Документация Narayana Billing API.

Текущая версия — 2.2

Введение

Любые приложения (в т.ч. клиентские и серверные) и любое оборудование, работающее в Narayana, взаимодействует с биллинговой системой (далее — биллингом) посредством API.

Для разграничения доступа к API используется система секретных ключей — идентификаторов клиентского и/или серверного оборудования и/или программного обеспечения, так или иначе взаимодействующего с биллингом.

Ключ может быть **клиентский** (требует обязательной авторизации пользователя) и **серверный с определенным уровнем доступа** (не требует авторизации). Клиентский ключ может быть как привязан к определенному пользователю (ключ с статической авторизацией) и игнорировать параметры login и password, переданные в запросе, так и требовать авторизацию (*ключ с динамической авторизацией*)

У любого ключа могут быть установлены следущие параметры:

- Название ключа (клиента) API
- Способ авторизации ключа
- Разрешенные IP-адреса
- Уровень доступа
- Пользователь по-умолчанию
- Разрешенные методы

Название ключа

В случае, если клиентом API является дочерняя телефонная станция, название клиента используется для привязки пользователей к АТС (SIP realm).

В противном случае название клиента используется для внутреннего учета и не влияет ни на что.

Способ авторизации

Способ авторизации определяет, по какому признаку будет производиться поиск пользователя при его авторизации. Возможным параметром может быть, к примеру, **username**, для авторизации по имени пользователя, **sip_extension**, для авторизации по SIP-логину, **id** для авторизации по внутреннему идентификационному номеру пользователя в базе данных.

Обращаем внимание, что желательно использовать уникальный параметр способа авторизации (как, например, имя пользователя). Использование потенциально неуникального параметра (к примеру, CallerID) может привести к непредсказуемым последствиям.

Разрешенные IP-адреса

Список IP-адресов, перечисленных через запятую, с которых могут производиться запросы к API. Допустимо использование масок. Пример: **178.1.1.*,87.1.1.2**

Уровень доступа

Используется для определения типа ключа, а так же присвоения ему определенного уровня доступа.

Уровень доступа 0 — клиентский ключ. Для данного типа ключа обязательна авторизация пользователя любым способом (динамичным с использованием пароля либо статичным), в противном случае исполнение не-анонимных методов будет недоступно.

Уровень доступа отличный от 0 — серверный ключ. Для данного ключа авторизация с использованием пароля не нужна, и признак пользователя, от имени которого исполняется запрос, передается параметром login. Однако следует помнить, что в случае, если владелец серверного ключа не является владельцем пользователя, найденного по переданному в запросе признаку login, система отклонит запрос (кроме случаев, когда ключ обладает исключительными правами.)

Пользователь по-умолчанию

Признак пользователя (к примеру, логин **username** или SIP-логин **sip_extension**, в зависимости от установленного способа авторизации), привязанный к ключу (статичная авторизация). Если данный параметр установлен, биллингом будут проигнорированы параметры **login** и **password**, переданные в запросе, и по-умолчанию будет авторизован пользователь, найденный по данному признаку.

Для того, чтобы произвести запрос от имени другого пользователя, принадлежащему владельцу ключа, следует передать параметр username.

Также, в некоторых случаях данный параметр используется для записи в память пользовательской переменной, необходимой для работы именно с этим ключом.

Для того, чтобы записать пользовательскую переменную в память, необходимо в переменную **config.api*custom*identities** конфигурационного файла добавить новый способ авторизации (к примеру, **payment*method*** *) и в параметрах ключа указать* ***payment*****method** как **<Способ авторизации>**, и значение, указанное как Пользователь, будет доступно как **request.payment_method**

Разрешенные методы

Список методов, доступных к исполнению для данного ключа, перечисленный через запятую. Если данный параметр не заполнен, разрешаются к исполнению любые методы.

Пример: **getInfo, getValue**

Запрос и ответ

1. Инициализация запроса к API

   URL запроса: **https://rdx.narayana.im/billing/api/${VERISON}/${APIKEY}/${METHOD}**

   Метод запроса: **POST** либо **GET**

2. Базовые параметры запроса

   - **login** — признак пользователя.
   - **password** — md5-хэш пароля пользователя (для динамической авторизации)
   - **username** — имя пользователя, от чьего имени производятся запросы к API (если не указано — используется текущий авторизованный пользователь)

3. Ответ API

   - Обычный ответ в формате JSON.

     Ответ на запрос представлен в виде JSON-массива с тремя переменными:

     {'result'} — результат исполнения метода (boolean)

     {'response'} — ответ либо код ошибки (string)

     {'description'} — текстовое описание причины ошибки (string)

     Пример:

     ```
     GET https://rdx.narayana.im/billing/api/2.2/key/getInfo?login=admin

     {"result":false,"response":"E_AUTH_ERROR"}
     ```

   - Текстовый ответ

     Используется для типов авторизации **sip_extension**, **id** и прочих указанных в конфигурационном файле (опция **config.api_plain_answer**).

     Содержит только текстовый ответ либо описание ошибки (**response**)

4. Ошибки

В результате некорректного запроса к API сервер может отдать следущие ошибки:

- **HTTP 400 Invalid Request** — некорректно сформирован запрос.
- **HTTP 401 Unauthorized** — некорректный ключ.
- **HTTP 403 Forbidden** — доступ с этого IP запрещен

NB! В случае запроса с некорректным и/или неактивным ключом, доступ с Вашего IP-адреса блокируется на неопределенный срок.

Коды ошибок

Ошибки инициализации запроса:

- **E_INCORRECT_API_VERSION** – Версия сервера и клиента не совпадают
- **E_UNKNOWN_METHOD** – Неизвестный метод
- **E_UNDEFINED_INSTRUCTION** – Невозможно выполнить метод
- **E_INSUFFICIENT_ACCESS** – Недостаточно прав для исполнения данного метода
- **E_MISSING_ARGUMENT** – Обязательный аргумент отсутствует
- **E_INVALID_ARGUMENT** – Аргумент неверного типа
- **E_LUA_EXCEPTION** – Критическая ошибка исполнения метода
- **E_AUTH_FAILED** – Неверная пара признак пользователя/пароль
- **E_HAS_NO_ONE_TOLD_YOU_SHE_IS_NOT_BREATHING** – Вход в вечный цикл. Вы никогда не должны видеть этой ошибки.

Ошибки исполнения запроса:

- **E_USER_LOCKED** – Пользователь заблокирован.
- **E_SQL_ERROR** – Ошибка SQL-запроса.
- **E_NOT_EXIST** – Запрошенный объект не существует в данном контексте.
- **E_DOES_NOT_BELONG_TO_YOU** – Запрошенный объект не принадлежит вам
- **E_DATA_ABORT** — Неверный ввод
- **E_ASTERISK_QUERY_ERROR** – Ошибка исполнения запроса к АТС.
- **E_SIP_SERVER_NOT_EXIST** – Указанный SIP-сервер не существует
- **E_WRONG_EXTENSION** – Недопустимый SIP логин
- **E_TRANSACTIONS_LIMITED** – Финансовые транзакции ограничены
- **E_TRANSACTIONS_LOCKED** – Финансовые транзакции заблокированы
- **E_ACCOUNT_HAS_USERS** – Невозможно удалить не пустой Лицевой счет
- **E_CANT_CREATE_ACCOUNT** – Невозможно создать Лицевой счет
- **E_CARRIER_ERROR** – Ошибка отправки SMS: канал вернул ошибку
- **E_DID_ORDER_FAIL** – Ошибка заказа DID-номера
- **E_INITIATE_CALL_FAIL** – Ошибка инициирования звонка
- **E_INSUFFICIENT_MONEY** – Недостаточно средств
- **E_INVALID_SMS_FROM** – Некорректное поле «Отправитель»
- **E_INVALID_SMS_TO** – Некорректное поле «Назначение»
- **E_CODE_HAS_NO_BALANCE** – На этом предоплаченном коде нет доступных средств
- **E_INVITE_LIMIT_REACHED** – Вы исчерпали лимит приглашений
- **E_MAXIMUM_PAYMENT** – Вы превысили максимальную сумму платежа
- **E_MINIMAL_PAYMENT** – Сумма платежа меньше минимальной
- **E_MINIMAL_START_PAYMENT** – Сумма первого платежа меньше минимальной
- **E_TARGET_WHITELISTED** – Данный IP-адрес находится в белом списке
- **E_TRANSACTION_FAILED** – Ошибка выполнения финансовой транзакции
- **E_TTS_ERROR** – Ошибка исполнения Text-to-Speech метода
- **E_UNROUTABLE** – Направление не обслуживается
- **E_CANT_CREATE_EXTENSION** — Невозможно создать учетную запись SIP
- **E_CANT_DELETE_EXTENSION** — Невозможно удалить учетную запись SIP
- **E_ALREADY_ON_THIS_TARIFF** — Вы уже обслуживаетесь на данном тарифном плане
- **E_CONTACT_RESELLER** – Данная операция недоступна пользователям реселлеров. Обратитесь к поставщику услуг.

Объект «Пользователь»

При работе с API биллинг-системы в основном используется объект user («Пользователь»).

В данном разделе описаны все атрибуты объекта и значения, которые они могут принимать.

Запросить значение какого-либо атрибута можно методами **getInfo()** (значения всех атрибутов), **getValue()** (значение конкретного атрибута)

Записать значения атрибутов можно соответствующими типу атрибута функциями:

- **updateUser()** — для данных пользователя
- **updateAccount()** — для данных лицевого счета
- **updateExtension()** — для данных учетной записи SIP.

Пояснения к типам:

- *string* — строковый тип;
- *int* — целочисленный тип;
- *float* — нецелое число;

Пользователь

- **id**(int) *[read only]* — внутренний идентификатор пользователя в системе.
- **username**(char[32]) *[read only]* — имя пользователя.
- **password**(char[32]) — зашифрованный пароль пользователя
- **manager**(char[32]) *[read only]* — владелец (менеджер) пользователя.
- **accesslevel**(int) — уровень доступа пользователя.
  - **10** — новый пользователь (ограничения — минимальный первый депозит)
  - **11** — пользователь
  - **22** — корпоративный пользователь (имеет доступ к управлению пользователями и тарифами)
  - **25** — партнер (имеет права корпоративного пользователя плюс доступ к основным маршрутам)
  - **33** — администратор системы
  - **44** — Мастер-администратор системы
- **active(int)** — состояние пользователя
  - **-1** — заблокирован
  - **0** — деактивирован (например, из-за нехватки баланса)
  - **1** — активен
  - **2** — блокировка внешних вызовов
  - **3** — блокировка всех вызовов
  - **99** — защита от блокировки
- **account**(int) — номер лицевого счета пользователя
- **currency**(char[3]) — код валюты пользователя
  - По-умолчанию — **EUR**
  - Доступные валюты можно получить посредством вызова метода **getCurrencies()**
- **language**(char[20]) — язык интерфейса пользователя
- **timezone**(float) — смещение часового пояса относительно UTC
- **allowed_rates**(char[20]) — список разрешенных тарифов для пользователя, перечисленных через запятую
- **allowed_call_len**(int) — максимально разрешенная длительность вызова для пользователя
- **notify_email**(char[256]) — электронная почта для уведомлений
- **notify_balance**(int) — отправлять уведомления на почту при достижении минимального баланса
  - **0** — не отправлять
  - **1** — отправлять
- **notify_balance_limit**(float) — лимит баланса для уведомления (в системной валюте)
- **notify_did**(int) — за какое количество дней до окончания действия аренды номера отправлять уведомление на почту
  - **-1** — не уведомлять
- **notify_ticket**(int) — отправлять уведомления на почту при получении ответа на тикет
  - **0** — не отправлять

    - **1** — отправлять
- **invites**(int) — количество доступных приглашений
- **comment**(char[1024]) — заметка о пользователе *(доступна только с уровнем доступа >= 33)*

Учетная запись SIP & DISA

- **disa_pin**(int) — PIN-код для доступа к DISA
- **disa_trusted_cli**(char[32]) — доверенный телефон для упрощенного доступа к DISA
- **sip_extension**(char[32]) — учетная запись SIP (5 или 7 знаков)
- **sip_password**(char[32]) — пароль SIP в открытом виде
- **sip_server**(char[128]]) — адрес SIP-сервера
- **sip_address**(char[128]]) — адрес SIP-сервера (DNS)
- **sip_callerid**(char[32]) — текущий идентификатор звонящего (АОН)
- **sip_language**(char[20]) — язык внутренних телефонных команд
- **sip_rates_id**(int) — идентификатор тарифного плана для этого SIP-логина
- **sip_rates_options**(char[128]) — идентификатор тарифных опций для этого SIP-логина, перечисленных через запятую
- **sip_srtp**(int) — шифрование SRTP
    - **0** — выключено
    - **1** — включено принудительно
- **sip_transport**(char[20]) — доступные транспорты для подключения к SIP-серверу
    - **udp,tcp,tls** — разрешить UDP / TCP / TLS
    - **udp,tcp** — разрешить UDP / TCP
    - **tls** — только TLS
    - **udp** — только UDP
    - **tcp** — только TCP
- **sip_max_call_len**(int) — максимальная длительность вызова для данного SIP-логина
- **sip_reach**(int) — доступность данного SIP-логина для звонка
    - **0** — недоступен
    - **2** — доступен
- **sip_redirect**(char[64]) — переадресовать входящие на SIP-логин звонки
- **sip_force_callback**(int) — принудительно обрывать вызов и создавать Callback на доверенный телефон
- **sip_realm**(char[16]) *[read only]* — название SIP-сервера
- **sip_pitch_shift**(int) — тон искажения голоса (-1 — выключено)
- **sip_host**(char[32]) — IP-адрес для авторизации по хосту (billing v2.3-rt)
- **sip_options**(char[128]) — дополнительные опции пира Asterisk (billing v2.3-rt)

Лицевой счет

- **balance**(float) — текущий баланс пользователя в системной валюте (EUR)
    - Пожалуйста, используйте метод **transferMoney()** для добавления средств пользователю.
- **overdraft**(float) — текущий кредитный лимит пользователя в системной валюте (EUR)
- **refill_allowed**(int) — разрешить проведения финансовых транзакций
    - **0** — финансовые операции запрещены
    - **1** — разрешены анонмные способы пополнения счета
    - **2** — разрешены все способы пополнения счета
- **postpaid**(int) — постоплатная система расчетов
    - **0** — выключено (используется предоплата)
    - **1** — включено (используется постоплата; раз в месяц выписывается счет)
- **billing_name**(char[100]) — полное имя (используется для выписывания счетов)
- **billing_address**(char[200]) — контактный адрес (используется для выписывания счетов)
- **billing_email**(char[100]) — электронная почта для пересылки счетов
- **billing_bank_account**(char[200]) — данные банковского счета для взаиморасчетов

Виртуальные атрибуты

Данные атрибуты пользователя, служащие для удобства обработки информации, доступны только для чтения, и не записаны в БД.

- **actualbalance** — актуальный баланс в системной валюте (EUR) с учетом овердрафта
- **displaycurrency** — название выбранной валюты (знак)
- **displaybalance** — текущий баланс в валюте пользователя
- **displayoverdraft** — кредитный лимит в валюте пользователя
- **displaynotifybalancelimit** — лимит баланса для уведомления в валюте пользователя
- **unread_tickets** — количество непрочитанных тикетов во внутренней тикет-системе

Функции API

Обязательные параметры выделены жирным цветом.
В случае, если необязательный параметр имеет значение по-умолчанию, оно указывается в [квадратных скобках]

Пояснения к типам:

- *string* — строковый тип;
- *ustring* — строковый тип со строгими значениями; допустимые значения указаны в {фигурных скобках};
- *num* — численный тип;
- *bool* — булевый тип (может принимать значения «true» или «false»);

В случае, если тип не указан — параметр может быть любого типа.

В (скобках) после названия метода указан необходимый уровень доступа для исполнения данного метода.
Методы с уровнем доступа *0* являются анонимными и могут быть исполнены без авторизации.

В примерах запросов к API используется версия API **2.2** и ключ «ffffffffff», а также опущены авторизационые данные.
Пожалуйста, убедитесь в том, что Вы используете актуальные данные перед исполнением запроса!

NB! За исключением раздела **«Работа с пользователем»** результаты в «Примерах запроса» будут указаны без

    {result: true, response: "something"}

а будут содержать в себе лишь значение **«response»**.

Если в поле «Возвращает» указано «*true* / *false*» — функция не имеет ответа, а имеет лишь результат исполнения **«result»** (*true* или *false*), но в то же время **«response»** может содержать код ошибки в случае неудачного исполнения.

Пользователь

getInfo(1) — Получение информации о пользователе.

Возвращает: JSON-массив с информацией о пользователе
Аргументы: —
Пример запроса:

```
GET /billing/api/2.2/ffffffffff/getInfo?login=admin
```

```
{"атрибут": "значение (см. раздел «Объект Пользователь»)",
"sip_extensions": [ {"атрибут_SIP_аккаунта": "значение (см. раздел «Объект Пользователь»)"}, ... ],
"did_numbers:" [ {"атрибут_DID_номера": "значение (см. функции «Прямые номера»"}, ... ],
"sim_cards:" [ {"атрибут_SIM_карты": "значение (см. функции «SIM-карты»"}, ...],
"broadcasts": [ {"атрибут_широковещательной_новости": "значение (см. функции «Новости и широковещательные сообщения»"}, ... ] }
```

getValue(1) — Получение конкретного параметра пользователя.

Возвращает: Значение запрашиваемого параметра Аргументы:

- **property**(string) — запрашиваемый параметр

Пример запроса:

```
GET /billing/api/2.2/fffffffff/getValue?property=balance
```

```
1.00
```

updateUser(1) — Изменение данных пользователя.

Возвращает: *true* / *false*
Аргументы:

- web_password(string) — новый пароль пользователя (md5-хэш)
- accesslevel(num) — уровень доступа
- active(num) — состояние
- account(num) — номер Лицевого счета
- currency(string) — код валюты
- language(string) — язык интерфейса
- timezone(num) — смещение часового пояса относительно UTC
- allowed_rates(string) — список разрешенных тарифов (через запятую)
- allowed_call_len(num) — максимально разрешенная длительность вызова (сек.)
- notify_email(string) — адрес e-mail для уведомлений
- notify_balance(num) — уведомлять при достижении минимального баланса
- notify_did(num) — уведомлять при достижении %i дней до конца срока аренды прямого номера
- notify_ticket(num) — уведомлять при ответах на тикет
- notify_balance_limit(num) — минимальный баланс для уведомлений
- invites(num) — количество доступных приглашений
- comment(string) — примечание

Пример запроса:

```
GET /billing/api/2.2/fffffffff/updateUser?username=test&active=0
```

```
true
```

Маршрутизация

getRoutes(22) — Получение списка доступных маршрутов.

Возвращает: JSON-массив со списком маршрутов и информацией о них.
Аргументы:

- update(bool) — обновить баланс запрашиваемых маршрутов *[false]*
- type(ustring{all,sip,sms,default}) — тип запрашиваемых маршрутов *[all]*
- all(bool) — запросить маршруты, принадлежащие корпоративным пользователям *[false]*

Пример запроса:

```
GET /billing/api/2.2/fffffffff/getRoutes?update=true
```

```
[{"id": ID маршрута,
"prefix":"SIP-prefix маршрута",
"cli": тип АОН (1 — только RU, 2 — международный, 99 — без ограничений),
"route":"название маршрута",
"owner":"владелец",
"enabled": статус (0 или 1),
"type":"тип (default, sms, sip)",
"lastupdate":"дата последнего обновления баланса",
"clientlastupdate":"клиентская дата последнего обновления баланса",
"balance":"баланс"}]
```

getRouteInfo(22) — Получение информации о маршруте по его ID

Возвращает: Информация о маршруте (JSON)
Аргументы:

- **id**(num) — идентификатор маршрута

Пример запроса:

```
GET /billing/api/2.2/fffffffff/getRouteInfo?id=1
```

```
{"id": ID маршрута,
"prefix":"SIP-prefix маршрута",
"cli": тип АОН (1 — только RU, 2 — международный, 99 — без ограничений),
"route":"название маршрута",
"owner":"владелец",
"enabled": статус (0 или 1),
"type":"тип (default, sms, sip)",
"lastupdate":"дата последнего обновления баланса",
"clientlastupdate":"клиентская дата последнего обновления баланса",
"balance":"баланс"}
```

addRoute(22) — Добавление нового маршрута в маршрутизацию

Возвращает: *true* / *false*
Аргументы:

- **route**(string) — название маршрута
- **type**(ustring{sip,sms}) — тип маршрута
- **sip_server**(string) — SIP-сервер для добавления маршрута
- **prefix**(string) — SIP-префикс для добавления маршрута (outlineN/) либо название SMS-обработчика
- host(string) — адрес хоста (домен либо IP), если необходимо *[]*
- user(string) — имя пользователя для авторизации *[]*
- fromuser(bool) — использовать имя пользователя как **fromuser** в настройках peer'а *[false]*
- pass(string) — пароль для авторизации *[]*
- callerid(num) — тип АОН (1 — только RU, 2 — международный, 99 — без ограничений) *[2]*
- balance_handler(string) — обработчик для проверки баланса *[]*
- balance_username(string) — имя пользователя для обработчика баланса *[]*
- balance_password(string) — пароль для обработчика баланса *[]*

Пример запроса:

```
GET /billing/api/2.2/fffffffff/addRoute?route=test&type=sms&sip_server=sip.serv.er&prefix=ourhandler
```

```
true
```

updateRoute(22) — Обновление параметров маршрута

Возвращает: *true* / *false*
Аргументы:

- **id**(num) — идентификатор маршрута
- route(string) — название маршрута
- prefix(string) — SIP-prefix (outlineN/) либо название SMS-обработчика
- host(string) — адрес хоста (домен либо IP)
- user(string) — имя пользователя для авторизации
- pass(string) — пароль для авторизации
- callerid(num) — тип АОН (1 — только RU, 2 — международный, 99 — без ограничений)
- balance_handler(string) — обработчик для проверки баланса
- balance_username(string) — имя пользователя для обработчика баланса
- balance_password(string) — пароль для обработчика баланса

Пример запроса:

```
GET /billing/api/2.2/fffffffff/updateRoute?id=1&route=changedName
```

```
true
```

updateRouteState(22) — Изменение состояния маршрута

Возвращает: *true* / *false*
Аргументы:

- **id**(num) — идентификатор маршрута
- **state**(bool) — состояние маршрута (true или false)

Пример запроса:

```
GET /billing/api/2.2/fffffffffff/updateRouteState?id=1&state=false
```

```
true
```

deleteRoute(22) — Удаление маршрута из маршрутизации

Возвращает: *true* / *false*
Аргументы:

- **id**(num) — идентификатор маршрута

Пример запроса:

```
GET /billing/api/2.2/fffffffffff/deleteRoute?id=1
```

```
true
```

Направления и тарификация

resolveDirection(0) — Разрешить направление по номеру телефона

Возвращает: Выбранную информацию о направлении
Аргументы:

- **number**(string) — номер телефона
- select(ustring{id,code,direction}) — Поле для вывода (ID направления, код или название) *[direction]*

Пример запроса:

```
GET /billing/api/2.2/fffffffffff/resolveDirection?number=79271871234
```

```
Russia Mobile - Megafon
```

getDirections(0) — Получение списка всех направлений

Возвращает: JSON-массив со списком всех направлений в системе
Аргументы:

- distinct(bool) — показывать только одно значение *code* для направления *[true]*

Пример запроса:

```
GET /billing/api/2.2/fffffffffff/getDirections
```

```
[{"code": код направления,
"id": ID направления,
"direction": "название направления",
"min_len": минимально допустимая длина номера,
"max_len": максимально допустимая длина номера}, ...]
```

getCurrentRates(0) — Получение текущей тарификации

Возвращает: JSON-массив в тарифами текущего пользователя
Аргументы:

- currency(string) — выбранная валюта *[EUR]*

Пример запроса:

```
GET /billing/api/2.2/fffffffffff/getCurrentRates
```

```
{"ID направления":
    {"direction":"Название направления",
        "routing":[
            {"routename":"Название маршрута",
            "route": ID маршрута,
            "displaycost": Цена в валюте пользователя,
            "priority": Приоритет,
            "cost": Цена в системной валюте
            }
        ]
    }
    ...
}
```

getRatesList(1) — Получение списка тарифных планов

Возвращает: JSON-массив — список доступных тарифных планов
Аргументы:

- all(bool) — Показать ТП, принадлежащие корпоративным пользователям *[false]*
- is_public(bool) — Показать только публичные тариыф *[false]*
- is_option(bool) — Показать тарифные опции (*true* — только тарифные опции, *false* — только тарифы, не установлено — показать все)

Пример запроса:

```
GET /billing/api/2.2/fffffffffff/getRatesList
```

```
[{"bill_type":"Тип тарификации",
"id":ID тарифа,
"is_public":Публичный ли тариф?,
"table_name":"Таблица в БД",
"owner":"Владелец тарифа",
"display_switch_price":Отображаемая цена подключения,
"name":"Название тарифа",
"switch_price":Цена подключение в системной валюте,
"did_multiplier":Множитель цен для DID-номеров,
"is_option":Тарифная опция?,
"multiplier":Множитель цен,
"description":"Описание тарифа"}, ...]
```

switchRates(1) — Смена тарифного плана, подключение и отключение тарифных опций

Возвращает: *true* / *false*
Аргументы:

- **id**(num) — ID тарифа (опции)
- state(bool) — Состояние (только для тарифных опций)
- sip_extension(string) — Учетная запись SIP для изменения

Пример запроса:

```
GET /billing/api/2.2/fffffffffff/switchRates?id=1&sip_extension=1000001
```

```
true
```

getRates(1) — Получение направлений в тарифе по ID тарифа (опции)

Возвращает: JSON-список направлений в тарифном плане
Аргументы:

- **id**(num) — ID тарифа (опции)

Пример запроса:

```
GET /billing/api/2.2/fffffffffff/getRates?id=1
```

```
{"ID направления":
    {"direction":"Название направления",
        "routing":[
            {"routename":"Название маршрута",
```

```
            "route": ID маршрута,
            "displaycost": Цена в валюте пользователя,
            "priority": Приоритет,
            "cost": Цена в системной валюте
            }
        ]
    }
    ...
}
```

getRatesInfo(22) — Получение параметров тарифного плана

Возвращает: Параметры тарифного плана (JSON)
Аргументы:

- **id**(num) — ID тарифа (опции)

Пример запроса:

```
GET /billing/api/2.2/fffffffff/getRatesInfo?id=1
```

```
{"bill_type":"Тип тарификации",
"id":ID тарифа,
"is_public":Публичный ли тариф?,
"table_name":"Таблица в БД",
"owner":"Владелец тарифа",
"display_switch_price":Отображаемая цена подключения,
"name":"Название тарифа",
"switch_price":Цена подключение в системной валюте,
"did_multiplier":Множитель цен для DID-номеров,
"is_option":Тарифная опция?,
"multiplier":Множитель цен,
"description":"Описание тарифа"}
```

addRates(22) — Добавление нового тарифного плана

Возвращает: *true* / *false*
Аргументы:

- **table**(string) — Название таблицы в БД
- **name**(string) — Название тарифного плана (опции)
- bill_type(string) — Тип тарификации *[by*minute]_
- is_option(bool) — Тарифная опция? *[false]*
- is_public(bool) — Публичный тариф? *[false]*
- switch_price(num) — Цена подключения *[0.00]*
- multiplier(num) — Множитель цен *[1.00]*
- did_multiplier(num) — Множитель цен прямых номеров *[1.00]*
- description(string) — Описание тарифа *[]*

Пример запроса:

```
GET /billing/api/2.2/fffffffff/addRates?table=test&name=test
```

```
true
```

updateRates(22) — Обновление параметров тарифного плана

Возвращает: *true* / *false*
Аргументы:

- **id**(num) — ID тарифа (опции)
- name(string) — Название тарифного плана (опции)
- bill_type(string) — Тип тарификации
- is_option(bool) — Тарифная опция?
- is_public(bool) — Публичный тариф?
- switch_price(num) — Цена подключения
- multiplier(num) — Множитель цен
- did_multiplier(num) — Множитель цен прямых номеров
- description(string) — Описание тарифа

Пример запроса:

```
GET /billing/api/2.2/fffffffff/updateRates?id=1&name=etst
```

```
true
```

deleteRates(22) — Удаление тарифного плана

Возвращает: *true* / *false*
Аргументы:

- **id**(num) — ID тарифа (опции)

Пример запроса:

```
GET /billing/api/2.2/fffffffff/deleteRates?id=1
```

```
true
```

addDirection(22) — Добавление направления в тарифный план

Возвращает: *true* / *false*
Аргументы:

- **rates**(num) — ID тарифного плана (опции)
- **direction**(num) — ID направления
- **cost**(num) — Цена за единицу тарификации
- **route**(num) — ID маршрута
- priority(num) — Приоритет *[1]*

Пример запроса:

```
GET /billing/api/2.2/fffffffff/addDirection?rates=1&direction=600&route=10&cost=0.15
```

```
true
```

updateDirection(22) — Обновление параметров направления в тарифном плане

Возвращает: *true* / *false*
Аргументы:

- **rates**(num) — ID изменяемого тарифного плана (опции)
- **direction**(num) — ID изменяемого направления
- prior(num) — Приоритет изменяемого направления *[1]*
- cost(num) — Цена за единицу тарификации
- route(num) — ID маршрута
- priority(num) — Приоритет

Пример запроса:

```
GET /billing/api/2.2/fffffffff/updateDirection?rates=1&direction=600&route=11&cost=0.18
```

```
true
```

updateDirections(22) — Обновление параметров направлений в тарифном плане

Возвращает: *true* / *false*
Аргументы:

- **rates**(num) — ID изменяемого тарифного плана (опции)
- **directions**(string) — JSON-массив со списком изменяемых направлений (формат см. в функции *updateDirection()*)

Пример запроса:

```
POST /billing/api/2.2/fffffffffff/updateDirections?rates=1
```

```
true
```

deleteDirection(22) — Удаление направления из тарифного плана

Возвращает: *true* / *false*
Аргументы:

- **rates**(num) — ID изменяемого тарифного плана (опции)
- **direction**(num) — ID удаляемого направления
- prior(num) — Приоритет удаляемого направления *[1]*

Пример запроса:

```
GET /billing/api/2.2/fffffffffff/deleteDirection?rates=1&direction=600
```

```
true
```

updateMarkup(22) — Автоматическая наценка в тарифном плане

Возвращает: *true* / *false*
Аргументы:

- **rates**(num) — ID изменяемого тарифного плана (опции)
- **markup**(num) — Множитель

Пример запроса:

```
GET /billing/api/2.2/fffffffffff/updateMarkup?rates=1&markup=1.2
```

```
true
```

copyDirections(22) — Копирование текущего тарифного плана пользователя в выбранный ТП

Возвращает: *true* / *false*
Аргументы:

- **rates**(num) — ID тарифного плана, куда будет скопирован текущий ТП

Пример запроса:

```
GET /billing/api/2.2/fffffffffff/copyDirections?rates=2
```

```
true
```

getBillTypes(22) — Получение списка доступных типов тарификации

Возвращает: JSON-массив — список доступных типов тарификации
Аргументы: —
Пример запроса:

```
GET /billing/api/2.2/fffffffffff/getBillTypes
```

```
[{"type":"Название типа",
"free_threshold": Нетарифицируемый порог,
"step":Шаг тарификации}, ...]
```

getBillType(33) — Получение информации о типе тарификации по его названию

Возвращает: Информация о типе тарификации (JSON)
Аргументы:

- **bill_type**(string) — Название типа тарификации

Пример запроса:

```
GET /billing/api/2.2/fffffffffff/getBillType?bill_type=by_second
```

```
{"type":"Название типа",
"free_threshold": Нетарифицируемый порог,
"step":Шаг тарификации}
```

addBillType(33) — Добавление типа тарификации

Возвращает: *true* / *false*
Аргументы:

- **bill_type**(string) — Название типа тарификации
- **free_threshold**(num) — Нетарифицируемый порог
- **step**(num) — Шаг тарификации (сек.)

Пример запроса:

```
GET /billing/api/2.2/fffffffffff/addBillType?bill_type=custom_type&free_threshold=3&step=10
```

```
true
```

updateBillType(33) — Обновление параметров типа тарификации

Возвращает: *true* / *false*
Аргументы:

- **bill_type**(string) — Название изменяемого типа тарификации
- free_threshold(num) — Нетарифицируемый порог
- step(num) — Шаг тарификации (сек.)

Пример запроса:

```
GET /billing/api/2.2/fffffffffff/updateBillType?bill_type=custom&free_threshold=0
```

```
true
```

deleteBillType(33) — Удаление типа тарификации

Возвращает: *true* / *false*
Аргументы:

- **bill_type**(string) — Название удаляемого типа тарификации

Пример запроса:

```
GET /billing/api/2.2/fffffffffff/deleteBillType?bill_type=custom
```

```
true
```

Обработка звонков

resolveCallerid(0) — Получение текущего CallerID пользователя **! DEPRECATED: Используйте getValue?property=sip_callerid !**

Возвращает: CallerID пользователя
Аргументы: —
Пример запроса:

```
GET /billing/api/2.2/ffffffffff/resolveCallerid
```

```
79010000001
```

createCallback(1) — Инициация Callback

Возвращает: *true* / *false*
Аргументы:

- callerid(string) — CallerID (в случае, если не указан — используем CallerID профиля)
- **destination1**(string) — Первое назначение
- **destination2**(string) — Второе назначение
- epitch(num) — Модификатор тона голоса (от 1 до 6)

Пример запроса:

```
GET /billing/api/2.2/ffffffffff/createCallback?destination1=79010000001&destination2=79020000002
```

```
true
```

callPrepare(50) — Инициация звонка

Возвращает: Allowed/Declined MaxCallTime/ErrorCode Destination CallerID DirectionName RoutePrefix RouteName PitchRx PitchTx
Аргументы:

- callerid(string) — CallerID (в случае, если не указан — используем CallerID профиля)
- **destination**(string) — Назначение
- prior(num) — Приоритет *[1]*
- from_originate(string) — Если звонок был инициирован не напрямую, в параметре необходимо указать **true**, либо DID-номер, на который пришел звонок
- source(string) — IP-адрес АТС, инициирующей звонок
- originated_by(string) — Инициатор звонка (например: user, system, slave и пр.)

Пример запроса:

```
GET /billing/api/2.2/ffffffffff/callPrepare?login=10000&destination=79010000001
```

```
Allowed 99999 79010000001 79020000002 RussiaMobile-Tele2 outline1/ route1-cli -1 -1
```

callFinish(50) — Завершение звонка

Возвращает: Saved DirectionName [CallerID/Destination] via [route] time cost CURRENCY SIPSTATUS
Аргументы:

- callerid(string) — CallerID (в случае, если не указан — используем CallerID профиля)
- **destination**(string) — Назначение
- prior(num) — Приоритет *[1]*
- **length**(num) — Длительность звонка в секундах
- status(string) — SIP-статус звонка
- is_did(string) — Если звонок был совершен на DID-номер, параметр принимает значение данного DID-номера

Пример запроса:

```
GET /billing/api/2.2/ffffffffff/callFinish?login=10000&destination=79020000002&length=0&status=0CANCEL
```

```
Saved RussiaMobile-Tele2 [79010000001/79020000002] via [route1-cli] 0 0 EUR 0CANCEL
```

getTextBalance(50) — Получение баланса пользователя в текстовом виде

Возвращает: Баланс для проговаривания АТС
Аргументы:

- balance(num) — Баланс для преобразования в текст (если не указано, используется баланс пользователя)
- balance(string) — Валюта (если не указано, используется валюта пользователя)

Пример запроса:

```
GET /billing/api/2.2/ffffffffff/getTextBalance?login=10000
```

```
ostatok&ноль&евро&ноль&центов&tadam3s
```

authorizeCall(50) — Авторизация пользователя в системе DISA

Возвращает: SIP extension авторизованного пользователя либо код ошибки (*Declined XXX*)
Аргументы:

- **callerid**(string) — CallerID, с которого поступил звонок на обработчик DISA
- **auth**(string) — Строка авторизации (обычно, это SIP extension + PIN-код)

Пример запроса:

```
GET /billing/api/2.2/ffffffffff/authorizeCall?callerid=79010000001&auth=100006666
```

```
10000
```

authorizeRealm(50) — Проверка подлинности запроса дочерней АТС

Возвращает: *true* / *false*
Аргументы:

- **source**(string) — IP-адрес дочерней АТС
- **auth**(string) — MD5-хэш от строки "${API-ключ}${IP-адрес АТС}${Имя API-ключа}"

Пример запроса:

```
GET /billing/api/2.2/ffffffffff/authorizeRealm?params
```

```
true
```

createTTS(50) — Преобразование текста в звуковой файл

Возвращает: аудиопоток в формате **wav**
Аргументы:

- **text**(string) — Текст

Пример запроса:

```
GET /billing/api/2.2/ffffffffff/createTTS?text=hello+world
```

```
<binary data>
```

Прямые номера

getDisaDIDList(0) — Получение списка номеров DISA

Возвращает: JSON-массив — список доступных номеров DISA
Аргументы: —
Пример запроса:

```
GET /billing/api/2.2/ffffffffff/getDisaDIDList
```

```
[{"did":"Номер доступа", "comment":"DISA\/Регион"}, ... ]
```

updateDIDState(1) — Изменение состояния прямого номера

Возвращает: *true* / _ false_
Аргументы:

- **did**(string) — Изменяемый DID-номер
- parameter(ustring{auto*renew, enabled}) — Изменяемый параметр _[auto*renew]_
- **state**(bool) — Состояние

Пример запроса:

```
GET /billing/api/2.2/fffffffff/updateDIDState?did=78005555550&parameter=enabled&state=false
```

```
true
```

getDIDPool(1) — Получение пула прямых номеров (список стран, локаций либо DID-номеров)

Возвращает: Пул прямых номеров
Аргументы:

- country(string) — Страна
- area(string) — Регион
- search_pattern(string) — Поисковый запрос

Примеры запросов:

```
GET /billing/api/2.2/fffffffff/getDIDPool
```

```
[{"country": "Страна"}, ...]
```

```
GET /billing/api/2.2/fffffffff/getDIDPool?country=Страна
```

```
[{"area": "Регион"}, ...]
```

```
GET /billing/api/2.2/fffffffff/getDIDPool?country=Страна&area=Регион&search_pattern=666
```

```
[{"displaycost": Цена аренды в валюте пользователя,
"displayinstallcost": Цена подключения в валюте пользователя,
"cost": Цена аренды в системной валюте,
"installcost": Цена подключения в системной валюте,
"country":"Страна",
"area":"Регион",
"voice_support":"Поддержка голоса",
"sms_support":"Поддержка SMS",
"did":"DID-номер",
"pool_id": ID пула,
"arg1": "Первый параметр для подключения",
"arg2": "Второй параметр для подключения",
"arg3": "Третий параметр для подключения"}, ...]
```

orderDID(1) — Заказ прямого номера

Возвращает: *true* / *false*
Аргументы:

- **pool_id**(num) — ID пула
- **did**(string) — прямой номер
- arg1(string) — Первый параметр
- arg2(string) — Второй параметр
- arg3(string) — Третий параметр

Пример запроса:

```
GET /billing/api/2.2/fffffffff/orderDID?pool_id=1&did=78005555550&arg1=something&arg2=somewhere
```

```
true
```

getDIDList(1) — Получение списка прямых номеров

Возвращает: JSON-массив — список прямых номеров
Аргументы:

- sort_by(ustring{expiration, cost, did) — Сортировка по (истечение аренды, стоимость, номер) *[expiration]*
- own(bool) — Показывать только собственные прямые номера *[true]*

Пример запроса:

```
GET /billing/api/2.2/fffffffff/getDIDList
```

```
[{"displaycost": Цена аренды в валюте пользователя,
"pool": ID пула,
"renew_notify": Отправлено уведомление о просрочке,
"expiration":"Дата истечения в формате ГГГГ-ММ-ДД ЧЧ:ММ:СС",
"enabled": Включен ли DID?,
"did": Прямой номер,
"auto_renew": Включено ли автопродление?,
"comment":"Комментарий",
"owner":"Владелец",
"cost":Цена аренды в системной валюте}, ...]
```

addDID(22) — Добавление прямого номера

Возвращает: *true* / *false*
Аргументы:

- **did**(string) — Прямой номер
- **owner**(string) — Владелец прямого номера
- comment(string) — Комментарий *[]*
- cost(num) — Цена аренды *[0.00]*
- expiration(string) — Дата истечения в формате ГГГГ-ММ-ДД ЧЧ:ММ:СС

Пример запроса:

```
GET /billing/api/2.2/fffffffff/addDID?did=78005555550&owner=user&cost=100
```

```
true
```

updateDID(22) — Обновление параметров прямого номера

Возвращает: *true* / *false*
Аргументы:

- **did**(string) — Изменяемый прямой номер
- owner(string) — Новый владелец
- comment(string) — Комментарий
- cost(num) — Цена аренды
- expiration(string) — Дата истечения в формате ГГГГ-ММ-ДД ЧЧ:ММ:СС

Пример запроса:

```
GET /billing/api/2.2/fffffffff/updateDID?did=78005555550&cost=101
```

```
true
```

deleteDID(22) — Удаление прямого номера

Возвращает: *true* / *false*
Аргументы:

- **did**(string) — Удаляемый прямой номер

Пример запроса:

```
GET /billing/api/2.2/fffffffff/deleteDID?did=78005555550
```

```
true
```

getDIDInfo(22) — Получение информации о прямом номере

Возвращает: Информация о прямом номере (JSON)
Аргументы:

- **did**(string) — Прямой номер

Пример запроса:

```
GET /billing/api/2.2/fffffffff/getDIDInfo?did=78005555550
```

```
{"displaycost": Цена аренды в валюте пользователя,
"expiration":"Дата истечения в формате ГГГГ-ММ-ДД ЧЧ:ММ:СС",
"owner":"Владелец",
"comment":"Комментарий",
"did": Прямой номер,
"auto_renew": Включено ли автопродление?,
"enabled": Включен ли прямой номер?,
"renew_notify": Отправлено уведомление о просрочке,
"cost": Цена аренды в системной валюте}
```

renewDID(22) — Продление прямого номера

Возвращает: *true* / *false*
Аргументы:

- **did**(string) — Продлеваемый DID-номер

Пример запроса:

```
GET /billing/api/2.2/fffffffff/renewDID?did=78005555550
```

```
true
```

addDIDPool(33) — Добавление пула прямых номеров

Возвращает: *true* / *false*
Аргументы:

- **country**(string) — Страна
- **area**(string) — Регион
- cost(num) — Цена аренды в месяц *[0.00]*
- installcost(num) — Цена подключения *[0.00]*
- provider(string) — Обработчик *[local]*
- user(string) — Пользователь (передается в обработчик) *[]*
- pass(string) — Пароль (передается в обработчик) *[]*
- param1(string) — Параметр 1 (передается в обработчик) *[]*
- param2(string) — Параметр 2 (передается в обработчик) *[]*
- param3(string) — Параметр 3 (передается в обработчик) *[]*
- qty(num) — Количество отображаемых номеров *[100]*

Пример запроса:

```
GET /billing/api/2.2/fffffffff/addDIDPool?country=Russia&area=Moscow&cost=50
```

```
true
```

updateDIDPool(33) — Обновление параметров пула прямых номеров

Возвращает: *true* / *false*
Аргументы:

- **id**(num) — ID изменяемого пула прямых номеров
- country(string) — Страна
- area(string) — Регион
- cost(num) — Цена аренды в месяц
- installcost(num) — Цена подключения
- provider(string) — Обработчик
- user(string) — Пользователь (передается в обработчик)
- pass(string) — Пароль (передается в обработчик)
- param1(string) — Параметр 1 (передается в обработчик)
- param2(string) — Параметр 2 (передается в обработчик)
- param3(string) — Параметр 3 (передется в обработчик)
- qty(num) — Количество отображаемых номеров

Пример запроса:

```
GET /billing/api/2.2/fffffffff/updateDIDPool?id=1&cost=100
```

```
true
```

getDIDPools(33) — Получение списка пулов прямых номеров

Возвращает: Список пула прямых номеров (JSON)
Аргументы: —
Пример запроса:

```
GET /billing/api/2.2/fffffffff/getDIDPools
```

```
[{"id": ID пула,
"country":"Страна",
"area":"Зона"
"cost": Цена в месяц,
"installcost": Цена установки,
"displayinstallcost": Цена установки в валюте пользователя,
"displaycost": Цена в месяц в валюте пользователя,
"quantity": Количество номеров на странице,
"provider":"Провайдер (обработчик)"
"username":"Имя пользователя для обработчика",
"password":"Пароль для обработчика",
"param1":"Параметр №1 для обработчика",
"param2":"Параметр №2 для обработчика",
"param3":"Параметр №3 для обработчика"}, ...]
```

getDIDPoolInfo(33) — Получение информации о пуле прямых номеров

Возвращает: Информация о пуле прямых номеров (JSON)
Аргументы:

- **id**(num) — ID пула прямых номеров

Пример запроса:

```
GET /billing/api/2.2/fffffffff/getDIDPoolInfo?id=1
```

```
{"id": ID пула,
"country":"Страна",
"area":"Зона"
"cost": Цена в месяц,
"installcost": Цена установки,
"displayinstallcost": Цена установки в валюте пользователя,
"displaycost": Цена в месяц в валюте пользователя,
"quantity": Количество номеров на странице,
"provider":"Провайдер (обработчик)"
"username":"Имя пользователя для обработчика",
"password":"Пароль для обработчика",
"param1":"Параметр №1 для обработчика",
"param2":"Параметр №2 для обработчика",
"param3":"Параметр №3 для обработчика"}
```

deleteDIDPool(33) — Удаление пула прямых номеров

Возвращает: *true* / *false*
Аргументы:

- **id**(num) — ID удаляемого пула прямых номеров

Пример запроса:

```
GET /billing/api/2.2/fffffffffff/deleteDIDPool?id=1
```

```
true
```

getDIDCount(50) — Проверка существования прямого номера

Возвращает: Количество найденных DID-номеров
Аргументы:

- did(string) — Искомый DID-номер

Пример запроса:

```
GET /billing/api/2.2/fffffffffff/getDIDCount?did=78005555550
```

```
1
```

Фильтры DIDTables

getDIDActions(1) — Получение списка возможных действий

Возвращает: Список возможных действий для DIDTables (JSON)
Аргументы: —
Пример запроса:

```
GET /billing/api/2.2/fffffffffff/getDIDActions
```

```
{"voice":
    {"Действие1":"Значение destination по-умолчанию"},
    {"Действие2":"Значение destination по-умолчанию"},
    ...
},
"sms":{
    {"Действие1 (SMS)":"Значение destination по-умолчанию"},
    {"Действие2 (SMS)":"Значение destination по-умолчанию"},
    ...
}
```

getFilters(1) — Получение списка возможных фильтров

Возвращает: Список возможных источников/паттернов для DIDTables (JSON)
Аргументы:

- resolver(string) — Получение списка доступных паттернов по конкретному обработчику
- sources(bool) — Получить только источники *[false]*

Пример запроса:

```
GET /billing/api/2.2/fffffffffff/getFilters
```

```
{"Источник1":{ "Паттерн1", "Паттерн2" },...}
```

addDIDRule(1) — Создание нового правила DIDTables

Возвращает: *true* / *false*
Аргументы:

- **did**(string) — Прямой номер
- source(string) — Используемый фильтр
- pattern(string) — Паттерн
- policy(ustring{ACCEPT,REJECT}) — Политика (принять/отклонить)
- sms_action(string) — Действие для обработки SMS-сообщения
- sms_destination(string) — Назначение (SMS)
- action(string) — Действие для обработки входящего звонка
- server(string) — Сервер, на котором выполняется правило
- destination(string) — Назначение
- callerid(string) — Передаваемый АОН (строка TRANSIT будет заменена на номер звонящего)
- timeout(num) — Таймаут (сек.)
- condition(string) — Условие (SUCCESS — успешный вызов, UNSUCCESS — неудачный вызов, BUSY — занят, NOANSWER — нет ответа, UNAVAIL — недоступен)
- input(string) — Ввод (если «условие» выбрано «Совпадение ввода»)
- position(ustring{start,end}) — Добавить правило в начало *[start]* или в конец *[end]*

Пример запроса:

```
GET /billing/api/2.2/fffffffffff/addDIDRule?did=78005555550&source=number&pattern=*&policy=ACCEPT&destination=10000
```

```
true
```

updateDIDRule(1) — Обновление правила DIDTables

Возвращает: *true* / *false*
Аргументы:

- **did**(string) — Прямой номер
- **rule**(num) — Номер изменяемого правила
- pos(num) — Новая позиция
- source(string) — Используемый фильтр
- pattern(string) — Паттерн
- policy(ustring{ACCEPT,REJECT}) — Политика (принять/отклонить)
- sms_action(string) — Действие для обработки SMS-сообщения
- sms_destination(string) — Назначение (SMS)
- action(string) — Действие для обработки входящего звонка
- server(string) — Сервер, на котором выполняется правило
- destination(string) — Назначение
- callerid(string) — Передаваемый АОН (строка TRANSIT будет заменена на номер звонящего)
- timeout(num) — Таймаут (сек.)
- condition(string) — Условие,
- input(string) — Ввод (если «условие» выбрано «Совпадение ввода»)

Пример запроса:

```
GET /billing/api/2.2/fffffffffff/updateDIDRule?did=78005555550&rule=1&pos=2
```

```
true
```

deleteDIDRule(1) — Удаление правила DIDTables

Возвращает: *true* / *false*
Аргументы:

- **did**(string) — Прямой номер
- **rule**(num) — Номер удаляемого правила
- pattern(string) — Паттерн (для удаления по паттерну — указать *rule* как *-1*)

Пример запроса:

```
GET /billing/api/2.2/fffffffffff/deleteDIDRule?did=78005555550&rule=-1&pattern=74955555550
```

```
true
```

getDIDRules(1) — Получение списка правил для прямого номера

Возвращает: Список правил для прямого номера (JSON)
Аргументы:

- **did**(string) — Прямой номер

- rule(num) — Номер правила (если не указано — возвращает все правила)

Пример запроса:

```
GET /billing/api/2.2/fffffffff/getDIDRules?did=78005555550
```

```
[{"timeout": Таймаут (с.),
"server":"Сервер, на котором выполняется правило",
"did": Прямой номер,
"policy":"Политика (принять/отклонить)",
"condition":"Условие","sms_action":"internal","source":"number","action":"playfile","pattern":"*","destination":"beep","rule":1,"callerid":"TRANSIT"},{"timeout":30,"server":"ptr-aa.narayana.im","did":78124099441,"policy":"ACCEF
```

resolveDIDRule(50) — Получение правил обработки правила DIDTables

Возвращает: OK ACCEPT/REJECT Владелец(extension) Действие(action) Сервер(sip*server) Назначение(destination) AOH(callerid) Таймаут(timeout) ДействиеSMS(sms*action) НазначениеSMS(sms_destination) — для выполнения правила
Возвращает: SKIP НомерПравила — для перехода на следущее доступное к выполнению правило
Возвращает: FIN — в случае, если правила закончились
Аргументы:

- **did**(string) — Прямой номер
- rule(num) — ID правила *[1]*
- **callerid**(string) — CallerID
- status(string) — SIP-статус
- input(string) — Ввод (SIP-статус должен быть передан как **99INPUT**)

Пример запроса:

```
GET /billing/api/2.2/fffffffff/resolveDIDRule?params
```

```
OK ACCEPT 10000 call rdx.narayana.im 10000 TRANSIT 30 internal 10000
```

Персональный план набора (алиасы)

getAliases(1) — Получение списка алиасов

Возвращает: Список алиасов (JSON)
Аргументы:

- all(bool) — Показать все алиасы (иначе — только свои алиасы)

Пример запроса:

```
GET /billing/api/2.2/fffffffff/getAliases
```

```
[{"global": Глобальный алиас (0/1),
"id": ID алиаса,
"owner":"Владелец алиаса",
"alias": Алиас,
"destination": Назначение}, ...]
```

getAliase(1) — Получение информации о алиасе

Возвращает: Информация о алиасе (JSON)
Аргументы:

- **id**(num) — ID алиаса

Пример запроса:

```
GET /billing/api/2.2/fffffffff/getAlias?id=11
```

```
{"global": Глобальный алиас (0/1),
"id": ID алиаса,
"owner":"Владелец алиаса",
"alias": Алиас,
"destination": Назначение}
```

addAlias(1) — Добавление нового алиаса

Возвращает: *true* / *false*
Аргументы:

- **alias**(string) — Алиас
- **destination**(string) — Назначение
- sip_extension(string) — SIP-логин, для которого устанавливается алиас
- global(bool) — Является ли алиас глобальным

Пример запроса:

```
GET /billing/api/2.2/fffffffff/addAlias?alias=111&destination=79010000001
```

```
true
```

updateAlias(1) — Обновление параметров алиаса

Возвращает: *true* / *false*
Аргументы:

- **id**(num) — ID алиаса
- destination(string) — Назначение
- sip_extension(string) — SIP-логин, для которого устанавливается алиас

Пример запроса:

```
GET /billing/api/2.2/fffffffff/updateAlias?alias=111&destination=79010000002
```

```
true
```

deleteAlias(1) — Удаление алиаса

Возвращает: *true* / *false*
Аргументы:

- **id**(num) — ID алиаса

Пример запроса:

```
GET /billing/api/2.2/fffffffff/deleteAlias?id=11
```

```
true
```

Безопасность

getBans(22) — Получение списка блокировок *iptables*

Возвращает: Список блокировок (JSON)
Аргументы: —
Пример запроса:

```
GET /billing/api/2.2/fffffffff/getBans
```

```
[{"reason":Причина блокировки,
"source":"Источник (хост) блокировки либо whitelist",
"id":ID блокировки,
"target":"Цель (хост) блокировки",
"date":"Дата блокировки",
"clientdate":"Дата блокировки в часовом поясе пользователя",
"permanent": Является ли бан постоянным}, ...]
```

flushBans(22) — Сброс блокировок на SIP-сервере пользователя

Возвращает: *true* / *false*
Аргументы: —
Пример запроса:

```
GET /billing/api/2.2/fffffffffff/flushBans
```

```
true
```

banIP(33) — Блокировка IP-адреса

Возвращает: *true* / *false*
Аргументы:

- **source**(string) — Источник (хост) блокировки либо whitelist
- **target**(string) — Цель (хост) блокировки
- reason(string) — Причина блокировки *[]*

Пример запроса:

```
GET /billing/api/2.2/fffffffffff/banIP?source=whitelist&target=127.0.0.1&reason=ya+je+localhost
```

```
true
```

unbanIP(33) — Разблокировка IP-адреса

Возвращает: *true* / *false*
Аргументы:

- **id**(num) — ID блокировки

Пример запроса:

```
GET /billing/api/2.2/fffffffffff/unbanIP?id=1
```

```
true
```

updateBanPermanent(33) — Обновление статуса блокировки

Возвращает: *true* / *false*
Аргументы:

- **id**(num) — ID блокировки
- **state**(bool) — Статус блокировки (*true* — постоянная, *false* — временная)

Пример запроса:

```
GET /billing/api/2.2/fffffffffff/updateBanPermanent?id=1&state=true
```

```
true
```

flushAllBans(33) — Сброс всех блокировок на всех серверах Narayana

Возвращает: *true* / *false*
Аргументы: —
Пример запроса:

```
GET /billing/api/2.2/fffffffffff/flushAllBans
```

```
true
```

closeChannels(50) — Закрыть все активные каналы пользователя

Возвращает: *true* / *false*
Аргументы: —
Пример запроса:

```
GET /billing/api/2.2/fffffffffff/closeChannels?login=10000
```

```
true
```

Предоплаченные коды (коды приглашения). Регистрация новых пользователей.

registerRequest(0) — Запрос на регистрацию нового пользователя / Генерация кода приглашения

Возвращает: Сгенерированный код приглашения
Аргументы:

- **signature**(string) — Уникальная подпись клиента (генерируется Web-сервером)

Пример запроса:

```
GET /billing/api/2.2/fffffffffff/registerRequest?signature=test
```

```
0000000000000000
```

registerByCode(0) — Регистрация нового пользователя по коду приглашения

Возвращает: *true* / *false*
Аргументы:

- **code**(string) — Инвайт-код (код предоплаченной карты)
- register(bool) — Произвести регистрацию (в противном случае — симуляция регистрации) *[false]*
- user(string) — Имя пользователя
- web_password(string) — MD5-хэш от пароля пользователя для входа в Web-интерфейс
- language(string) — Язык Web-интерфейса

Пример запроса:

```
GET /billing/api/2.2/fffffffffff/registerByCode?code=0000000000000000&register=true&user=test&web_password=098f6bcd4621d373cade4e832627b4f6&language=nenglish
```

```
true
```

activatePrepaidCode(1) — Активация кода приглашения (кода предоплаченной карты) на счет пользователя

Возвращает: *true* / *false*
Аргументы:

- **code**(string) — Код приглашения
- **amount**(num) — Сумма к зачислению (в случае, если указано *0* — зачисляем весь доступный остаток)

Пример запроса:

```
GET /billing/api/2.2/fffffffffff/activatePrepaidCode?code=0000000000000000&amount=0
```

```
true
```

getPrepaidCodes(1) — Получение списка сгенерированных кодов приглашений (кодов предоплаченных карт)

Возвращает: Список сгенерированных кодов приглашения (JSON)

Аргументы:

- active(bool) — Показывать только неактивированные коды *[false]*
- all(bool) — Показать все коды (иначе — только коды, созданные лично вами) *[false]*
- count(bool) — Показать только количество кодов *[false]*

Пример запроса:

```
GET /billing/api/2.2/fffffffffff/getPrepaidCodes
```

```
[{"created":"Дата создания",
"clientcreated": "Дата создания в часовом поясе пользователя",
"balance": Баланс,
"displaybalance": Баланс в валюте пользователя,
"code":"Код",
"accesslevel": Уровень доступа,
"owner":"Создатель кода",
"activated_to":"Кем активирован",
"activated_date":"Дата активации"
"client_activated_date":"Дата активации в часовом поясе пользователя"}, ...]
```

createPrepaidCode(1) — Генерация нового кода приглашения (предоплаченного кода)

Возвращает: Сгенерированный код приглашения
Аргументы:

- accesslevel(num) — Уровень доступа *[10]*
- balance(num) — Баланс *[0.00]*
- target(string) — Пользователь либо SIM-карта, который может активировать данный код (если не указано — без ограничений)_ *[]*

Пример запроса:

```
GET /billing/api/2.2/fffffffffff/createPrepaidCode?balance=10.00
```

```
true
```

deletePrepaidCode(1) — Удаление неактивированного кода приглашения (кода предоплаченной карты)

Возвращает: *true* / *false*
Аргументы:

- **code**(string) — Код приглашения

Пример запроса:

```
GET /billing/api/2.2/fffffffffff/deletePrepaidCode?code=0000000000000000
```

```
true
```

Сообщения

receiveSMS(0) — Получение нового сообщения от внешнего источника

Возвращает: *true* / *false*
Аргументы:

- **msg_handler**(string) — Идентификатор обработчика (обычно привязывается к API-ключу, поэтому, указывать не обязательно)

Пример запроса:

```
GET /billing/api/2.2/fffffffffff/receiveSMS?from=18005555550&to=78005555550&text=Hello+world
```

```
true
```

receiveDeliveryReport(0) — Получение отчета о доставке от внешнего источника

Возвращает: *true* / *false*
Аргументы:
* **msg_handler**(string) — Идентификатор обработчика (обычно привязывается к API-ключу, поэтому, указывать не обязательно)
Пример запроса:

```
GET /billing/api/2.2/fffffffffff/receiveDeliveryReport?id=AAAAAAAAAAAAAAAAAA&status=DELIVERED
```

```
true
```

msgCallback(0) — Инициация обратного вызова с помощью SMS от внешнего источника

Возвращает: *true* / *false*
Аргументы:

- **msisdn**(string) — Номер телефона, с которого пришел запрос на инициацию
- **text**(string) — Текст запроса в формате "SIPEXTEN PIN [CALLERID] DESTINATION1 [DESTINATION2]" (в квадратных скобках указаны необязательные параметры)
- **to**(string) — Номер телефона, на который пришел запрос на инициацию

Пример запроса:

```
GET /billing/api/2.2/fffffffffff/msgCallback?msisdn=78005555550&to=18005555550&text=40000+1111+74999999999+79111111111
```

```
true
```

readSMS(1) — Отметить SMS-сообщения как прочитанные

Возвращает: *true* / *false*
Аргументы:

- **status**(num) — Отмечать только сообщения с данным статусом
- **did**(string) — Отмечать сообщения только для данного прямого номера

Пример запроса:

```
GET /billing/api/2.2/fffffffffff/readSMS?status=0&did=18005555550
```

```
true
```

sendSMS(1) — Отправить SMS-сообщение

Возвращает: Уникальный идентификатор сообщения
Аргументы:

- callerid(string) — Отображаемый номер (если не указан — берется из профиля пользователя)
- **to**(string) — Номер телефона назначения
- **text**(string) — Текст сообщения (urlencoded)

Пример запроса:

```
GET /billing/api/2.2/fffffffffff/sendSMS?to=18005555550&text=Hello+World!
```

```
uniqueid
```

sendHLR(1) — Отправить HLR-запрос

Возвращает: Результат HLR-запроса OK,IMSI,MCC,MNC,LAC
Аргументы: —

- **to**(string) — Номер телефона для HLR-запроса

Пример запроса:

```
GET /billing/api/2.2/ffffffffff/sendHLR?to=18005555550
```

```
OK,999881111111111,999,888,000000
```

sendVoiceMessage(1) — Отправить голосовое сообщение

Возвращает: Уникальный идентификатор сообщения
Аргументы:

- callerid(string) — Отображаемый номер (если не указан — берется из профиля пользователя)
- **to**(string) — Номер телефона назначения
- **text**(string) — Текст сообщения (urlencoded) либо ссылка на аудиофайл

Пример запроса:

```
GET /billing/api/2.2/ffffffffff/sendVoiceMessage?callerid=18005555550&to=780055555550&text=hello,+world
```

```
uniqueid
```

sendSIPMessage(1) — Отправить SIP-сообщение абоненту сервиса

Возвращает: *true* / *false*
Аргументы:

- callerid(string) — Отображаемый номер (если не указан — берется из профиля пользователя)
- **to**(string) — Номер телефона назначения
- **text**(string) — Текст сообщения (urlencoded)

Пример запроса:

```
GET /billing/api/2.2/ffffffffff/sendVoiceMessage?to=40000&text=hello+world
```

```
true
```

deliveryReport(1) — Получить отчет о доставке уникальному ID сообщения

Возвращает: SMS Sent (отправлено) / SMS Deliv (доставлено) / SMS Undeliv (не доставлено)
Аргументы:

- **string**(num) — ID записи в журнале вызовов

Пример запроса:

```
GET /billing/api/2.2/ffffffffff/deliveryReport?id=uniqueid
```

```
SMS Deliv
```

setDeliveryReport(95) — Установить отчет о доставке для SMS-сообщения

Возвращает: *true* / *false*
Аргументы:

- **id**(num) — Идентификатор сообщения
- **status**(num) — Статус

Пример запроса:

```
GET /billing/api/2.2/ffffffffff/setDeliveryReport?id=1&status=0
```

```
true
```

getSMS(95) — Получение списка всех сообщений **! DEPRECATED: Используйте readSMS() !**

Возвращает: Список SMS-сообщений (JSON)
Аргументы:

- **status**(num) — Статус для получения сообщений Пример запроса:

```
GET /billing/api/2.2/ffffffffff/getSMS?status=0
```

```
[{"id": идентификатор сообщения (для setDeliveryReport),
"date": "дата получения сообщения (server time)",
"src": "отправитель",
"dest": "назначение",
"msg": "сообщение",
"status": статус (0 — непрочитано, 1 — прочитано),
"type": "тип (in — входящее, out — исходящее)"}, ...]}
```

Управление пользователями

getUsers(22) — Показать список пользователей

Возвращает: Список пользователей (JSON)
Аргументы:

- all(bool) — показать пользователей, принадлежащих корпоративным пользователям *[false]*

Пример запроса:

```
GET /billing/api/2.2/ffffffffff/getUsers
```

```
[{"атрибут": "значение"}, ...]
Список аттрибутов см. в описании функции _getInfo()
```

registerUser(22) — Создать нового пользователя

Возвращает: *true* / *false*
Аргументы:

- **user**(string) — имя пользователя
- **web_password**(string) — md5-хэш от пароля пользователя
- **accesslevel**(num) — уровень доступа
- **sip_server**(string) — SIP-сервер
- **sip_extension**(string) — SIP-логин
- **sip_password**(string) — SIP-пароль
- balance(num) — Начальный баланс в валюте регистрирующего пользователя *[0.00]*
- language(string) — Язык интерфейса пользователя *[config.default*language]_
- currency(string) — Валюта пользователя *[EUR]*
- sip_srtp(bool) — Использовать шифрование SRTP *[false]*
- account(string) — Номер лицевого счета (или auto для автоматического создания) *[auto]*

! NB: даже в случае, если указана валюта регистрируемого пользователя, в качестве начального баланса будет использована валюта регистрирующего пользователя.

Пример запроса:

```
GET /billing/api/2.2/ffffffffff/registerUser?user=newuser&web_password=5f4dcc3b5aa765d61d8327deb882cf99&accesslevel=11&sip_server=sip.server.com&sip_extension=40000&sip_password=verysecretpassword&balance=10.00
```

```
true
```

updateUser(1) — Обновление параметров пользователя

Возвращает: *true* / *false*
Аргументы:

- user(string) — Логин пользователя
- web_password(string) — Новый пароль для авторизации (md5-хэш)
- accesslevel(num) — Уровень доступа

- active(num) — Состояние (1 — активен, 0 — неактивен, -1 — заблокирован)
- account(num) — Номер лицевого счета
- currency(string) — Валюта пользователя
- language(string) — Язык интерфейса пользователя
- timezone(num) — Часовой пояс (смещение от UTC)
- rates(num) — Идентификатор тарифного плана
- allowed_rates(string) — Разрешенные к подключению тарифный план
- allowed_call_len(num) — Максимальная длительность вызова
- notify_email(string) — E-Mail для уведомлений
- notify_balance(num) — Получать уведомления при снижении баланса до мин. порога
- notify_did(num) — Количество дней до отключения DID-номера для уведомлений
- notify_ticket(num) — Получать уведомления при ответе на тикеты
- notify_balance_limit(num) — Минимальный порог баланса для уведомлений
- invites(num) — Количество приглашений
- comment(string) — Комментарий

Пример запроса:

```
GET /billing/api/2.2/fffffffff/updateUser?user=newuser&active=-1
```

```
true
```

deleteUser(22) — Удалние пользователя

Возвращает: *true* / *false*
Аргументы:

- **user**(string) — Логин пользователя

Пример запроса:

```
GET /billing/api/2.2/fffffffff/deleteUser?user=newuser
```

```
true
```

Учетные записи SIP

getExtensions(1) — Получение списка SIP-аккаунтов пользователя

Возвращает: Список SIP-аккаунтов пользователя (JSON)
Аргументы:

- own(bool) — Получить только свои SIP-аккаунты *[true]*
- all(bool) — Получить список всех SIP-аккаунтов *[false]*
- owner(string) — Получить список SIP-аккаунтов указанного пользователя

Пример запроса:

```
GET /billing/api/2.2/fffffffff/getExtensions?own=true
```

```
[{"sip_transport":"Транспорт SIP (udp,tcp,tls/udp/tcp/tls/udp,tcp)",
"disa_pin": PIN-код,
"sip_callerid": "Идентификатор звонящего (АОН)",
"owner":"Владелец SIP-аккаунта",
"sip_password":"SIP-пароль",
"sip_host":"Хост для авторизации по IP",
"sip_srtp": Шифрование SRTP (0/1),
"sip_language":"Язык SIP-аккаунта (Язык внутренних телефонных команд)",
"sip_reach": Доступность абонента для вызова (2 – доступен, 0 – недоступен),
"sip_redirect": Переадресовать входящие звонки сюда,
"sip_extension": SIP-логин,
"sip_realm":"Краткое название SIP-сервера",
"disa_trusted_cli": "Доверенный CallerID",
"rates_name":"название тарифного плана",
"sip_max_call_len": Максимальная длительность вызова,
"sip_address":"Адрес SIP-сервера" (new in v2.3),
"sip_rates_id": идентификатор тарифного плана,
"sip_rates_options": тарифные опции, перечисленные через запятую,
"sip_server":"IP-адрес SIP-сервера",
"lastupdated":"Последнее обновление статуса" (для realtime),
"sip_force_callback": Принудительный Callback (0/1),
"sip_pitch_shift": Искажение голоса (-1 – выключено)}, ...]
```

getExtensionInfo(1) — Получение информации о SIP-аккаунте

Возвращает: Информация о SIP-аккаунте (JSON) либо значение запрошенного параметра
Аргументы:

- **sip_extension**(string) — SIP-аккаунт для получения информации
- property(string) — запрашиваемый параметр для получения значения конкретного параметра

Пример запроса:

```
GET /billing/api/2.2/fffffffff/getExtensionInfo?sip_extension=40000
```

```
{"sip_transport":"Транспорт SIP (udp,tcp,tls/udp/tcp/tls/udp,tcp)",
"disa_pin": PIN-код,
"sip_callerid": "Идентификатор звонящего (АОН)",
"owner":"Владелец SIP-аккаунта",
"sip_password":"SIP-пароль",
"sip_host":"Хост для авторизации по IP",
"sip_srtp": Шифрование SRTP (0/1),
"sip_language":"Язык SIP-аккаунта (Язык внутренних телефонных команд)",
"sip_reach": Доступность абонента для вызова (2 – доступен, 0 – недоступен),
"sip_redirect": Переадресовать входящие звонки сюда,
"sip_extension": SIP-логин,
"sip_realm":"Краткое название SIP-сервера",
"disa_trusted_cli": "Доверенный CallerID",
"rates_name":"название тарифного плана",
"sip_max_call_len": Максимальная длительность вызова,
"sip_address":"Адрес SIP-сервера" (new in v2.3),
"sip_rates_id": идентификатор тарифного плана,
"sip_rates_options": тарифные опции, перечисленные через запятую,
"sip_server":"IP-адрес SIP-сервера",
"lastupdated":"Последнее обновление статуса" (для realtime),
"sip_force_callback": Принудительный Callback (0/1),
"sip_pitch_shift": Искажение голоса (-1 – выключено)}
```

getAvailExtension(1) — Получение ближайшего доступного для регистрации SIP-аккаунта

Возвращает: Доступный SIP аккаунт для регистрации
Аргументы:

- own(bool) — Вернуть следущий доступный дополнительный (пользовательский) SIP аккаунт для регистрации (XXXXXyy) *[false]*

Пример запроса:

```
GET /billing/api/2.2/fffffffff/getAvailExtension?own=true
```

```
4000001
```

createExtension(1) — Создание учетной записи SIP

Возвращает: *true* / *false*
Аргументы:

- owner(string) — Владелец SIP-аккаунта
- **sip_extension**(string) — SIP-аккаунт
- **sip_password**(string) — Пароль SIP
- sip_server(string) — Адрес SIP сервера
- sip_callerid(string) — Определяемый номер (АОН) по-умолчанию
- sip_language(string) — Язык SIP-аккаунта по-умолчанию
- sip_srtp(bool) — Использовать шифрование SRTP *[false]*

Пример запроса:

```
GET /billing/api/2.2/fffffffff/createExtension?sip_extension=4000001&sip_password=password
```

```
true
```

updateExtension(1) — Изменение SIP-аккаунта

Возвращает: *true* / *false*
Аргументы:

- sip_extension(string) — SIP-аккаунт для изменения
- extension(string) — Новое название (номер) SIP-аккаунта
- sip_password(string) — Новый пароль SIP
- sip_server(string) — Новый SIP сервер
- sip_srtp(num) — Использовать шифрование SRTP (0/1)
- sip_transport(ustring) — Разрешенный SIP транспорт (udp,tcp,tls/udp/tcp/tls/udp,tcp)
- sip_callerid(string) — Определяемый номер (АОН)
- sip_max_call_len(num) — Максимальная длительность вызова
- sip_reach(num) — Доступность абонента для вызова (2 — доступен, 0 — недоступен)
- sip_redirect(string) — Куда переадресовывать входящие звонки на данный SIP-логин
- sip_force_callback(num) — Принудительный Callback на доверенный номер (**disa*trusted*cli**) (0/1)
- sip_language(string) — Язык внутренних телефонных команд
- sip_rates_id(num) — Идентификатор тарифного плана
- disa_pin(num) — PIN-код
- disa_trusted_cli(string) — Доверенный телефон

Пример запроса:

```
GET /billing/api/2.2/fffffffff/updateExtension?sip_extension=40000&sip_srtp=1&sip_reach=2
```

```
true
```

deleteExtension(1) — Удаление учетной записи SIP

Возвращает: *true* / *false*
Аргументы:

- **sip_extension**(string) — SIP-аккаунт для удаления

Пример запроса:

```
GET /billing/api/2.2/fffffffff/deleteExtension?sip_extension=40000
```

```
true
```

Журнал событий

getLogs(1) — Получение журнала событий

Возвращает: Журнал событий (JSON)
Аргументы:

- limit(num) — Получить последние N записей *[100]*
- only_completed(bool) — Показать только успешные вызовы *[false]*
- resolve_info(bool) — Показать дополнительную информацию о вызываемом абоненте *[false]*
- user(string) — пользователь для получения журнала вызовов

Пример запроса:

```
GET /billing/api/2.2/fffffffff/getLogs?resolve_info=true
```

```
[{"direction":"Направление",
"route":"Маршрут",
"cost": Цена,
"date":"Дата записи",
"displaycost": Цена в валюте пользователя,
"id": ID записи,
"username":"Имя пользователя",
"displaycurrency":"Валюта пользователя",
"status":"Статус",
"clientdate":"Дата с часовым поясом пользователя",
"destination":"Назначение",
"time":"Длительность вызова",
"callerid":"АОН",
"sip_extension": "SIP-логин",
"info": "Дополнительная информация о вызываемом абоненте",
"debug": "Отладочная информация о звонке"}, ...]
```

deleteLogs(1) — Удаление журнала событий

Возвращает: *true* / *false*
Аргументы:

- user(string) — Имя пользователя для удаления журнала (если не указан — удаляем собственный журнал)

Пример запроса:

```
GET /billing/api/2.2/fffffffff/deleteLogs
```

```
true
```

Функции интерфейса

getCourse(0) — Получение внутреннего курса валют и прочих данных

Возвращает: Курс валют (JSON)
Аргументы:

- **currency**(string) — Валюта для получения инфо

Пример запроса:

```
GET /billing/api/2.2/fffffffff/getCourse?currency=EUR
```

```
{"eur_course": Курс к Евро,"currency":"Код валюты","displayname":"€","transfer_fee":Стомость перевода между ЛС,"referral_reward":Вознаграждение за приглашение}
```

getCurrencies(0) — Получение списка валют

Возвращает: Список валют (JSON)
Аргументы: —
Пример запроса:

```
GET /billing/api/2.2/fffffffff/getCurrencies
```

```
[{"eur_course":1,"currency":"EUR","displayname":"€"},{"eur_course":68.917,"currency":"RUB","displayname":"руб. "}]
```

Новости и широковещательные сообщения

getNews(0) — Получение новостей

Возвращает: Список новостей или конкретная новость (JSON)
Аргументы:

- id(string) — Идентификатор новости для получения
- all(bool) — Показать все новости
- broadcast(bool) — Показать только новости с широковещательным сообщением

Пример запроса:

```
GET /billing/api/2.2/fffffffff/getNews
```

```
[{"is_broadcast": широковещательное? (0/1),
"id":"идентификатор новости",
```

```
"author":"автор новости",
"message":"Тестовое содержание",
"access": минимальный уровень доступа для просмотра,
"language":"язык для просмотра",
"lastupdated":"посленее обновление (часовой пояс сервера)",
"clientlastupdated":"последнее обновление (часовой пояс пользователя)",
"broadcast":"Широковещательное сообщение",
"date":"дата (часовой пояс сервера)",
"clientdate":"дата (часовой пояс клиента)",
"header":"Тестовое сообщение",
"lastupdated_by":"автор последнего обновления",
"typ":"тип (влияет только на отображение; warning/error/success/info)"}, ... ]
```

addNews(33) — Добавление новости

Возвращает: *true* / *false*
Аргументы:

- **id**(string) — Идентификатор новости
- is*broadcast(bool) — Широковещательная новость? _[false]*
- user(string) — Кто может читать эту новость (если не указана — все могут читать новость)
- access(num) — Минимальный уровень доступа (если 0 — даже неавторизованные пользователи) *[0]*
- language(string) — Язык пользователей, которые могут читать новость (если не указан — могут читать пользователи с любым языком)
- typ(ustring{info,warning,error,success}) — Тип (используется для отображение в web-интерфейсе) *[info]*
- broadcast(string) — Широковещательное сообщение
- header(string) — Заголовок новости
- message(string) — Текст новости

Пример запроса:

```
GET /billing/api/2.2/ffffffffff/addNews?id=testnews&header=test&message=hello+world
```

```
true
```

updateNews(33) — Изменение новости

Возвращает: *true* / *false*
Аргументы:

- **id**(string) — Идентификатор новости
- is*broadcast(bool) — Широковещательная новость? _[false]*
- user(string) — Кто может читать эту новость (если не указана — все могут читать новость)
- access(num) — Минимальный уровень доступа (если 0 — даже неавторизованные пользователи) *[0]*
- language(string) — Язык пользователей, которые могут читать новость (если не указан — могут читать пользователи с любым языком)
- typ(ustring{info,warning,error,success}) — Тип (используется для отображение в web-интерфейсе) *[info]*
- broadcast(string) — Широковещательное сообщение
- header(string) — Заголовок новости
- message(string) — Текст новости

Пример запроса:

```
GET /billing/api/2.2/ffffffffff/updateNews?id=testnews&message=hello+earth
```

```
true
```

deleteNews(33) — Удаление новости

Возвращает: *true* / *false*
Аргументы:

- **id**(string) — Идентификатор новости

Пример запроса:

```
GET /billing/api/2.2/ffffffffff/deleteNews?id=testnews
```

```
true
```

Тикет-система

getUnreadTickets(1) — Получение количества непрочитанных сообщений **! DEPRECATED: функция больше не используется !**

Возвращает: Количество непрочитанных тикетов
Аргументы: —
Пример запроса:

```
GET /billing/api/2.2/ffffffffff/getUnreadTickets
```

```
1
```

createTicket(1) — Создать тикет или ответить в существующий

Возвращает: Номер созданного / существующего тикета
Аргументы:

- **destination**(string) — Логин получателя
- **subject**(string) — Тема
- **message**(string) — Сообщение
- comment(string) — Комментарий (только для администраторов)
- ticket(num) — Номер тикета для ответа в него

Пример запроса:

```
GET /billing/api/2.2/ffffffffff/createTicket?destination=support&subject=Help&message=Phone+is+not+ringing!
```

```
5162
```

getTickets(1) — Получение списка тикетов

Возвращает: Список тикетов (JSON)
Аргументы:

- limit(num) — Получить N последних тикетов *[800]*
- all(bool) — Получить все тикеты *[false]*

Пример запроса:

```
GET /billing/api/2.2/ffffffffff/getTickets
```

```
[{"status": статус тикета (1 — непрочитан, 0 — прочитан, -1 — закрыт),
"unread": непрочитан (true/false),
"subject":"тема",
"sender":"отправитель",
"recipient":"Dahog",
"ticket": номер тикета,
"msgid": номер сообщения,
"date":"дата в часовом поясе сервера",
"clientdate":"дата в часовом поясе сервера"}, ... ]
```

getTicket(1) — Получение тикета

Возвращает: Сообщения тикета (JSON)
Аргументы:

- **ticket**(num) — Номер тикета

Пример запроса:

```
GET /billing/api/2.2/ffffffffff/getTicket?params
```

```
[{"type":"тип сообщения (in/out)",
```

```
"ticket": номер тикета,
"reply":"пользователь для ответа",
"message":"сообщение",
"status":статус (0 — прочитано, 1 — непрочитано, -1 — закрыто),
"subject":"тема",
"sender":"отправитель",
"username":"логин отправителя (для тех случаев, если sender == support)",
"date":"дата (часовой пояс сервера)",
"clientdate":"дата (часовой пояс клиента)",
"recipient":"получатель",
"msgid": идентификатор сообщения}, ... ]
```

updateTicket(1) — Обновление статуса тикета

Возвращает: *true* / *false*
Аргументы:

- **ticket**(num) — Номер тикета
- action(ustring{close,open,mark*unread,assign*to) — Действие (close — закрыть, open — открыть, mark*unread — сделать непрочитанным) _[open]*
- assign_to(string) — Список пользователей (перечисленных через запятую), кому назначить данный тикет

Пример запроса:

```
GET /billing/api/2.2/fffffffffff/updateTicket?ticket=5162&action=mark_unread
```

```
true
```

deleteTicket(33) — Удаление тикета

Возвращает: *true* / *false*
Аргументы:

- **ticket**(num) — Номер тикета
- username(string) — Пользователь (для удаления всех его тикетов)

Пример запроса:

```
GET /billing/api/2.2/fffffffffff/deleteTicket?ticket=5162
```

```
true
```

deleteMessage(33) — Удаление конкретного сообщения

Возвращает: *true* / *false*
Аргументы:

- **id**(num) — Идентификатор сообщения

Пример запроса:

```
GET /billing/api/2.2/fffffffffff/deleteMessage?id=10000
```

```
true
```

Финансы и статистика

getPaymentMethods(0) — Получение списка доступных методов оплаты

Возвращает: Список доступных методов оплаты (JSON)
Аргументы: —
Пример запроса:

```
GET /billing/api/2.2/fffffffffff/getPaymentMethods
```

```
{"private": {"метод оплаты для верифицированных пользователей":"Название Метода"},
"public":{"метод оплаты для всех пользователей":"Название метода"},
"default": "метод оплаты по-умолчанию"}
```

finishInvoice(0) — Запрос от обработчика на завершение оплаты

Возвращает: *зависит от обработчика*
Аргументы: *зависит от обработчика*

Пример запроса:

```
POST /billing/api/2.2/fffffffffff/finishInvoice
```

```
ok
```

getStats(1) — Получение статистики

Возвращает: Статистика (JSON)
Аргументы: —

- financials(bool) — Получение финансовой информации *[false]*
- statistic(bool) — Получение статистики *[false]*
- date1(string) — Дата (от)
- date2(string) — Дата (до)

Пример запроса:

```
GET /billing/api/2.2/fffffffffff/getStats
```

```
{"forecast": {
  "forecast": на сколько дней пользователю хватит денежных средств,
  "recommmend": рекомендуемый платеж в системной валюте (EUR),
  "display_recommend": рекомендуемый платеж в валюте пользователя},
"financials": {
  "gsm_cost": роуминговые расходы в системной валюте (EUR),
  "display_gsm_cost": роуминговые расходы в валюте пользователя,
  "call_cost": расходы на звонки в системной валюте (EUR),
  "display_call_cost": расходы на звонки в валюте пользователя,
  "sms_cost": расходы на SMS в системной валюте (EUR),
  "display_sms_cost": расходы на SMS в валюте пользователя,
  "gprs_cost": расходы на мобильный интернет в системной валюте (EUR),
  "display_gprs_cost": расходы на мобильный интернет в валюте пользователя,
  "other_cost": прочие расходы (в т.ч. переводы) в системной валюте (EUR)
  "display_other_cost": прочие расходы (в т.ч. переводы) в валюте пользователя,
  "total_cost": общие расходы в системной валюте (EUR),
  "display_total_cost": общие расходы в валюте пользователя},
"stat":{
  "total_calls": всего звонков,
  "success_calls": успешных звонков,
  "success_calls_percent": процент успешных звонков,
  "calls_duration": общая длительность вызовов,

  "total_sms": всего SMS-сообщений,
  "success_sms": успешных SMS-сообщений,
  "success_sms_percent": процент доставленных SMS-сообщений,

  "total_gsm_calls": всего GSM-вызовов,
  "success_gsm_calls": успешных GSM-вызовов,
  "success_gsm_calls_percent": процент успешных GSM-вызовов,
  "gsm_calls_duration": длительность GSM-вызовов,

  "gprs_traffic": получено/передано трафика (моб.интернет) Кбайть}
}
```

transferMoney(1) — Перевод средств на счет другого пользователя

Возвращает: *true* / *false*
Аргументы:

- **destination**(string) — Получатель (логин или номер лицевого счета)

- **amount**(num) — Сумма в валюте запрашивающего пользователя
- comment(string) — Комментарий

Пример запроса:

```
GET /billing/api/2.2/ffffffffff/transferMoney?destination=100000000&amount=100&comment=Enjoy
```

```
true
```

createInvoice(1) — Создать счет для оплаты

Возвращает: Информация для оплата счета (JSON)
Аргументы: —

- **payment_method**(string) — Обработчик оплаты
- **payment_type**(string) — Способ оплаты
- **payment_amount**(num) — Сумма в валюте запрашивающего пользователя

Пример запроса:

```
GET /billing/api/2.2/ffffffffff/createInvoice?payment_method=pscb&payment_type=qiwi
```

```
{"comission": комиссия (процент),
"transaction": номер транзакции (инвойса),
"msg": "произвольное сообщение о особенностях оплаты",
"method": "метод запроса в URL (GET/POST)",
"url": "URL, по которому нужно перейти для оплаты инвойса",
-- Внимание: последущие параметры нужно передать в указанный URL указанным в «method» способом! --
"параметр1": "значение1",
"параметр2": "значнение2"
```

generateInvoice(1) — Генерация PDF-инвойса по указанным данным **! DEPRECATED: Используйте viewInvoice(invoice) !**

Возвращает: PDF-инвойс
Аргументы:

- **invoice**(num) — Номер инвойса
- **date**(string) — Дата выставления
- **deadline**(string) — Дата
- **services**(string) — Услуги в формате Услуга@Единиц тарифицировано@Название единицы@Стоимость единицы,Услуга2@Единиц@Название единицы@Стоимость,..

Пример запроса:

```
GET /billing/api/2.2/ffffffffff/generateInvoice?invoice=1&date=2017-06-24 00:00:00&deadline=2017-06-25 00:00:00&services=Service@1@Unit@100
```

```
%PDF-1.4\n1 0 obj\n<< BINARY PDF FILE
```

monthlyInvoice(1) — Генерация счета (инвойса) за один месяц

Возвращает: PDF-инвойс Аргументы:

- month(num) — Номер месяца для генерации инвойса (используется прошлый, если не указано)
- email(bool) — Прислать инвойс на электронную почту *[false]*

Пример запроса:

```
GET /billing/api/2.2/ffffffffff/monthlyInvoice
```

```
%PDF-1.4\n1 0 obj\n<< BINARY PDF FILE
```

getInvoices(1) — Получить список инвойсов

Возвращает: Список инвойсов (JSON)
Аргументы:

- own(bool) — Показать только свои инвойсы *[true]*
- bank(bool) — Показать только банковские переводы *[false]*
- done(bool) — Показать только завершенные инвойсы

Пример запроса:

```
GET /billing/api/2.2/ffffffffff/getInvoices?params
```

```
[{"id": номер инвойса,
"owner":"владелец инвойса (кому выставлен)",
"comission": комиссия,
"vat": НДС,
"amount": сумма в системной валюте (EUR),
"displayamount": сумма в валюте пользователя,
"status": статус,
"msg":"Произвольное сообщение о особенностях оплаты",
"method":"метод оплаты",
"handler": "обработчик (системная информация)",
"billing_data":"ФИО, адрес пользователя",
"txdata": "системная информация о транзакции (см. функцию _createInvoice()_)",
"date":"дата выставления инвойса (время сервера)",
"clientdate":"дата выставления инвойса (время с часовым поясом клиента)",
"validthru":"инвойс валиден до (время сервера)",
"clientvalidthru": "инвойс валиден до (время с часовым поясом клиента)}, ... ]
```

getInvoiceInfo(1) — Получение информации о инвойсе

Возвращает: Информация о инвойсе (JSON)
Аргументы:

- **invoice**(num) — Номер инвойса

Пример запроса:

```
GET /billing/api/2.2/ffffffffff/getInvoiceInfo?id=1
```

```
{"id": номер инвойса,
"owner":"владелец инвойса (кому выставлен)",
"comission": комиссия,
"vat": НДС,
"amount": сумма в системной валюте (EUR),
"displayamount": сумма в валюте пользователя,
"status": статус,
"msg":"Произвольное сообщение о особенностях оплаты",
"method":"метод оплаты",
"handler": "обработчик (системная информация)",
"billing_data":"ФИО, адрес пользователя",
"txdata": "системная информация о транзакции (см. функцию _createInvoice()_)",
"date":"дата выставления инвойса (время сервера)",
"clientdate":"дата выставления инвойса (время с часовым поясом клиента)",
"validthru":"инвойс валиден до (время сервера)",
"clientvalidthru": "инвойс валиден до (время с часовым поясом клиента)}
```

viewInvoice(1) — Загрузить PDF-инвойс

Возвращает: Бинарный PDF-файл
Аргументы:

- **invoice**(num) — Номер инвойса

Пример запроса:

```
GET /billing/api/2.2/ffffffffff/viewInvoice?params
```

```
%PDF-1.4\n1 0 obj\n<< BINARY PDF FILE
```

deleteInvoice(33) — Удаление инвойса

Возвращает: *true* / *false*
Аргументы:

- **invoice**(num) — Номер инвойса

Пример запроса:

```
GET /billing/api/2.2/fffffffffff/deleteInvoice?invoice=1
```

```
true
```

Лицевые счета

updateAccount(1) — Изменение параметров лицевого счета

Возвращает: *true* / *false*
Аргументы:

- account(num) — Номер лицевого счета
- balance(num) — Баланс в валюте **запрашивающего** пользователя
- overdraft(num) — Кредитный лимит в валюте **запрашивающего** пользователя
- refill_allowed(num) — Финансовые операции разрешены (0 — запрещены, 1 — разрешены, 2 — верифицирован)
- billing_name(string) — ФИО для выставления счетов
- billing_address(string) — Адрес для выставления счетов
- billing_email(string) — Электронный адрес для выставления счетов
- billing_bank_account(string) — Номер банковского счета для получения и выставления счетов
- postpaid(num) — Постоплатный аккаунт (присылать счет в конце месяца) (0/1)

! NB: пользователи с уровнем доступа выше 33 (администраторы) не могут менять баланс пользователей! Используйте функцию transferMoney()

Пример запроса:

```
GET /billing/api/2.2/fffffffffff/updateAccount?account=100000000&postpaid=1
```

```
true
```

getAccounts(22) — Получение списка лицевых счетов пользователей

Возвращает: Списо лицевых счетов и информация по ним (JSON)
Аргументы:

- all(bool) — Показать лицевые счета клиентов реселлеров *[false]*

Пример запроса:

```
GET /billing/api/2.2/fffffffffff/getAccounts
```

```
[ {"account": номер лицевого счета,
"users":["список", "пользователей", "на лицевом счете"],
"balance": баланс в системной валюте (EUR),
"displaybalance": баланс в валюте запрашивающего пользователя,
"overdraft": кредитный лимит в системной валюте,
"displayoverdraft": кредитный лимит в валюте запрашивающего пользователя,
"displaycurrency":"€ ($/руб./и т.д.)",
"refill_allowed": финансовые операции разрешены (0 — запрещены, 1 — разрешены, 2 — верифицирован),
"postpaid": Постоплатный аккаунт (присылать ли счета в конце месяца) }, ... ]
```

getAccountInfo(22) — Получение информации о лицевом счете

Возвращает: Информация о лицевом счете (JSON)
Аргументы:

- **account**(num) — Номер лицевого счета

Пример запроса:

```
GET /billing/api/2.2/fffffffffff/getAccountInfo?account=100000000
```

```
{"account": номер лицевого счета,
"users":["список", "пользователей", "на лицевом счете"],
"balance": баланс в системной валюте (EUR),
"displaybalance": баланс в валюте запрашивающего пользователя,
"overdraft": кредитный лимит в системной валюте,
"displayoverdraft": кредитный лимит в валюте запрашивающего пользователя,
"displaycurrency":"€ ($/руб./и т.д.)",
"refill_allowed": финансовые операции разрешены (0 — запрещены, 1 — разрешены, 2 — верифицирован),
"postpaid": postpaid-аккаунт (присылать ли счета в конце месяца) }
```

deleteAccount(22) — Удаление лицевого счета

Возвращает: *true* / *false*
Аргументы:

- **account**(num) — Номер лицевого счета

Пример запроса:

```
GET /billing/api/2.2/fffffffffff/deleteAccount?account=100000000
```

```
true
```

Доступ к API

getKey(33) — Получение данных API-ключа

Возвращает: Информацию о API-ключе (JSON)
Аргументы:

- **name**(string) — Идентфикатор API-ключа

Пример запроса:

```
GET /billing/api/2.2/fffffffffff/getKey?name=TESTKEY1
```

```
{"maintenance":"Дата последнего тех.обслуживания (для системных целей)",
"auth_by":"Авторизация по",
"dnsname":"DNS-имя (для SIP-серверов)",
"source":"IP-адрес или маска (*.*.*.*)",
"name":"Идентификатор ключа",
"secure_only": Только HTTPS (0/1),
"enabled": Активен (0/1),
"accesslevel": Уровень доступа,
"key":"API-ключ"}
```

getKeys(33) — Получение списка

Возвращает: Список API-ключей (JSON)
Аргументы: —
Пример запроса:

```
GET /billing/api/2.2/fffffffffff/getKeys
```

```
[{"maintenance":"Дата последнего тех.обслуживания (для системных целей)",
"auth_by":"Авторизация по",
"dnsname":"DNS-имя (для SIP-серверов)",
"source":"IP-адрес или маска (*.*.*.*)",
"name":"Идентификатор ключа",
"secure_only": Только HTTPS (0/1),
"enabled": Активен (0/1),
"accesslevel": Уровень доступа,
"key":"API-ключ"}, ...]
```

addKey(33) — Создание нового API-ключа

Возвращает: *true* / *false*
Аргументы:

- **name**(string) — Идентификатор API-клиента
- **auth_by**(string) — Авторизация по (id,username,sip_extension,msisdn,iccid...)
- **source**(string) — IP-адрес (или маска)
- **apikey**(string) — API ключ
- **accesslevel**(num) — Уровень доступа (0 для клиентских ключей)
- default_user(string) — Привязанный пользователь
- allowed_methods(string) — Разрешенные к исполнению методы
- secure_only(num) — Запросы только по *https://*

Пример запроса:

```
GET /billing/api/2.2/fffffffff/addKey?params
```

```
answer example
```

updateKey(33) — Изменение данных API-ключа

Возвращает: *true* / *false*
Аргументы:

- **name**(string) — Идентификатор API-клиента
- auth_by(string) — Авторизация по (id,username,sip_extension,msisdn,iccid...)
- source(string) — IP-адрес (или маска)
- apikey(string) — API ключ
- accesslevel(num) — Уровень доступа (0 для клиентских ключей)
- default_user(string) — Привязанный пользователь
- allowed_methods(string) — Разрешенные к исполнению методы
- secure_only(num) — Запросы только по *https://*
- enabled(num) — Статус (1 — активен, 0 — выключен)

Пример запроса:

```
GET /billing/api/2.2/fffffffff/updateKey?name=TESTKEY1&secure_only=1
```

---

revokeKey(33) — Отзыв API-ключа

Возвращает: *true* / *false*
Аргументы:

- **name**(string) — Идентификатор API-клиента

Пример запроса:

```
GET /billing/api/2.2/fffffffff/revokeKey?name=TESTKEY1
```

```
true
```

GSM

gsmRequest(0) — Запрос к GSM-обработчику

Возвращает: *зависит от обработчика*
Аргументы:

- **provider**(string) — Имя обработчика (размещенного в директории **inc/gsm/**)
- action(string) — Название субпрограммы для вызова (в ином случае — вызывается Default)

Пример запроса:

```
POST /billing/api/2.2/fffffffff/gsmRequest
```

```
ok
```

SIM-карты

getSIMList(1) — Получение списка SIM-карт

Возвращает: Список SIM-карт (JSON)
Аргументы:

- all(bool) — Показать все доступные SIM-карты (в том числе, неактивные) *[true]*
- own(bool) — Показать только SIM-карты, принадлежащие текущему пользователю *[false]*

Пример запроса:

```
GET /billing/api/2.2/fffffffff/getSIMList
```

```
[{"allow_direct_sms": Разрешить прямые SMS в обход сети Narayana,
"msisdn": MSISDN (номер абонента),
"owner":"Владелец SIM-карты",
"lastnetworkdate":"Дата последней регистрации в сети",
"iccid":"ICCID SIM-карты",
"puk1":PUK1-код,
"puk2":PUK2-код,
"callback_cid":CallerID для Callback,
"sip_extension":SIP-профиль,
"provider":"Провайдер (обработчик)",
"enabled": Состояние SIM-карты (1/0),
"lastnetwork":MCCMNC сети}, ...]
```

getSIMInfo(1) — Получение информации о SIM-карте

Возвращает: Информация о SIM-карте (JSON)
Аргументы:

- **iccid**(string) — ICCID SIM-карты

Пример запроса:

```
GET /billing/api/2.2/fffffffff/getSIMInfo?iccid=89372
```

```
{"allow_direct_sms": Разрешить прямые SMS в обход сети Narayana,
"msisdn": MSISDN (номер абонента),
"owner":"Владелец SIM-карты",
"lastnetworkdate":"Дата последней регистрации в сети",
"iccid":"ICCID SIM-карты",
"puk1":PUK1-код,
"puk2":PUK2-код,
"callback_cid":CallerID для Callback,
"sip_extension":SIP-профиль,
"provider":"Провайдер (обработчик)",
"enabled": Состояние SIM-карты (1/0),
"lastnetwork":MCCMNC сети}
```

bindSIM(1) — Инициация процесса привязки SIM-карты

Возвращает: *true* / *false*
Аргументы:

- **iccid**(string) — ICCID SIM-карты
- **puk1**(string) — PUK1-код
- **puk2**(string) — PUK2-код

Пример запроса:

```
GET /billing/api/2.2/fffffffff/bindSIM?iccid=89372&puk1=1111&puk2=2222
```

```
true
```

unbindSIM(1) — Инициация процесса отвязки SIM-карты

Возвращает: *true* / *false*
Аргументы:

- **iccid**(string) — ICCID SIM-карты

Пример запроса:

```
GET /billing/api/2.2/ffffffffff/unbindSIM?iccid=89372
```

```
true
```

updateSIM(1) — Обновление параметров SIM-карты

Возвращает: *true* / *false*
Аргументы:

- **iccid**(string) — ICCID SIM-карты
- owner(string) — Владелец SIM-карты
- puk1(string) — PUK1-код
- puk2(string) — PUK2-код
- sip_extension(string) — Используемый SIP-профиль
- callback_cid(string) — CallerID для Callback
- directsms(num) — Разрешить прямые SMS в обход сети Narayana (0 — нет, 1 — да)
- enabled(num) — Активация SIM-карты (0 — выключена, 1 — включена)

Пример запроса:

```
GET /billing/api/2.2/ffffffffff/updateSIM?iccid=89372&sip_extension=12345
```

```
true
```

addSIM(33) — Добавление SIM-карты

Возвращает: *true* / *false*
Аргументы:

- **provider**(string) — Провайдер (обработчик)
- **msisdn**(string) — MSISDN (абонентский номер)
- **iccid**(string) — ICCID/IMSI (уникальный номер SIM-карты)
- puk1(string) — PUK1-код (необходим для активации)
- puk2(string) — PUK2-код (необходим для активации)

Пример запроса:

```
GET /billing/api/2.2/ffffffffff/addSIM?provider=test&msisdn=111&iccid=89372&puk1=0000&puk2=0000
```

```
true
```

deleteSIM(33) — Удаление SIM-карты

Возвращает: *true* / *false*
Аргументы:

- **iccid**(string) — ICCID SIM-карты

Пример запроса:

```
GET /billing/api/2.2/ffffffffff/deleteSIM?iccid=89372
```

```
true
```

notifySIM(50) — Отправка SMS-сообщения на SIM-карту

Возвращает: *true* / *false*
Аргументы:

- **sim**(string) — ICCID SIM-карты
- **from**(string) — Номер отправителя
- **text**(string) — Текст сообщения

Пример запроса:

```
GET /billing/api/2.2/ffffffffff/notifySIM?sim=89372&from=111&text=hello+world
```

```
true
```

Техническое обслуживание

asteriskRequest(22) — Запрос к АТС

Возвращает: Ответ АТС
Аргументы:

- **asterisk**(string) — IP-адрес АТС
- **destination**(string) — Назначение запроса (к примеру, **interface**)
- **process**(string) — Первый аргумент (обычно, название метода)
- params(string) — Параметры запроса

Пример запроса:

```
GET /billing/api/2.2/ffffffffff/asteriskRequest?asterisk=127.0.0.1&destination=sipreload&process=sipreload
```

```
ok
```

updateDialplan(33) — Обновление версии диалплана

Возвращает: OK <версия dialplan>
Аргументы:

- **asterisk**(string) — IP-адрес АТС
- **url**(string) — URL файла dialplan, где слэш **/** заменен тильдой **~**

Пример запроса:

```
GET /billing/api/2.2/ffffffffff/updateDialplan?asterisk=127.0.0.1&url=localhost~dialplan
```

```
OK 0082R20170626_205
```

restoreDialplan(33) — Откат версии диалплана к предыдущей

Возвращает: OK <версия dialplan>
Аргументы:

- **asterisk**(string) — IP-адрес АТС

Пример запроса:

```
GET /billing/api/2.2/ffffffffff/restoreDialplan
```

```
OK 0082R20170626_205
```

updateInterface(33) — Обновление файла *interface*

Возвращает: *true* / *false*
Аргументы:

- **asterisk**(string) — IP-адрес АТС
- **url**(string) — URL файла interface, где слэш **/** заменен тильдой **~**

Пример запроса:

```
GET /billing/api/2.2/fffffffff/updateInterface?asterisk=127.0.0.1&url=localhost~interface
```

```
OK 0.12.4
```

doMaintenance(90) — Провести регулярные процедуры в БД (курсы валют, инвойсы, DID-номера и пр.)

Возвращает: *true* / *false*
Аргументы: Пример запроса:

```
GET /billing/api/2.2/fffffffff/doMaintenance
```

```
true
```