Программный модуль «Бизнес.API»

Руководство пользователя

Редакция от 22.10.2020

Содержание

Сокращения

ПМ - программный модуль.

Бизнес.Ру - Платформа "Бизнес.Ру".

Назначение

Это руководство содержит информацию по работе с ПМ „Бизнес.API“ для сотрудника предприятия.

ПМ имеет два интерфейса, связанных между собой через серверную часть - веб-интерфейс и программный интерфейс.

Веб-интерфейс доступен из веб-браузера в аккаунте Бизнес.Ру для авторизованных пользователей. Пример адреса: https://action_295919.business.ru/ . Язык интерфейса - русский.

Программный интерфейс доступен из любого стороннего веб-приложения для авторизованных пользователей. Пример адреса: https://action_295919.business.ru/api/rest/customerorders.json

ПМ предназначен для индивидуальных предпринимателей и юридических лиц, осуществляющих продажу товаров и услуг.

Сведения о правообладателях

ПМ разработан ООО „Класс Информационные Технологии“. Авторские прав на ПМ принадлежат ООО „Класс Информационные Технологии“.

Авторизация запросов к API Бизнес.Ру

При отправке запроса к API Бизнес.Ру приложение должно выполнить ряд действий:

1. К массиву параметров запроса добавляется параметр app_id (Id интеграции из формы настройки интеграции), инициализированный значением, сохраненным в базе данных приложения.
2. Полученный массив параметров сортируется в алфавитном порядке и преобразуется в url-кодированную строку.
3. Вычисляется пароль запроса по формуле:

app_psw = MD5(token + secret + params_string),

где параметром функции MD5 служит конкатенация компонентов:
token - сохраненный в базе данных приложения токен;
secret - предоставленный приложению секретный ключ (Секретный ключ из формы настройки интеграции);
params_string - url-кодированная строка параметров.
4. К массиву параметров запроса добавляется параметр app_psw, инициализированный рассчитанным ранее значением.
5. Необходимый запрос с параметрами отправляется по адресу, сохраненному в базе данных приложения.

Получив запрос, сервис Бизнес.Ру выполняет следующее:
1. Авторизует запрос - проверяет наличие параметров app_id и app_psw, получает строку параметров без app_psw, вычисляет MD5(token + secret + params_string), используя токен и секретный ключ, сохраненные в базе данных для интеграции, идентифицируемой параметром app_id, Если полученное значение совпадает со значением параметра запроса app_psw, запрос считается авторизованным.
2. В соответствие с запросом, осуществляет необходимые манипуляции с данными аккаунта, в процессе чего получает результирующий массив.
3. Перед тем, как вернуть результирующий массив в приложение, сервис Бизнес.Ру генерирует новый токен, сохраняет его в своей базе данных, ассоциируя с текущей интеграцией, присоединяет к результирующему массиву, после чего преобразует результирующий массив (с новым токеном) в JSON-строку и вычисляет пароль по формуле:
app_psw = MD5(token + secret + result_json),
где параметром функции MD5 служит конкатенация компонентов:
token - старый токен;
secret - предоставленный приложению секретный ключ;
result_json - JSON-строка результата.
4. К результирующему массиву присоединяется полученный пароль, после чего результирующий массив в формате JSON возвращается в приложение в составе HTTP-ответа с кодом статуса 200 OK.

Получив результат, приложение выполняет следующее:
1. Осуществляет проверку результата - преобразует JSON-строку результата в массив, проверяет наличие элементов массива token и app_psw, создает копию массива без app_psw, на основе которой получает JSON-строку и вычисляет MD5(token + secret + result_json), используя сохраненное в базе данных приложения значение токена. Если

полученное значение совпадает со значением элемента массива app_psw, считается, что проверка пройдена.
2. В базе данных старое значение токена перезаписывается новым.
3. Результат запроса, без token и app_psw обрабатывается прикладной логикой приложения.

Секретный ключ, известный сервису Бизнес.Ру и приложению, но никогда не передаваемый по сети, а также постоянно обновляемые сервисом токены, с помощью которых формируются пароли, являющиеся функцией передаваемых данных, исключают возможность их злоумышленной подмены третьей стороной. Секретный ключ следует хранить в тайне.

Для интеграций по API для каждой пары значений "id интеграции+Секретный ключ" время жизни токена равно 1 часу. Метод repair библиотеки business_ru_api.php в течение 1 часа после получения токена будет возвращать тот же ранее полученный токен. Данное решению позволяет существенно упростить работу с параллельными запросами к API Бизнес.Ру.

Потеря токена и его восстановление

Потеря токена может произойти в случае, если сервис Бизнес.Ру, получив от приложения запрос, успешно его обработал, сгенерировал и записал в свою базу данных новый токен, отправил его в составе результата приложению, но оно по какой-то причине результат не получило. В этом случае приложение отправит следующий запрос, в котором пароль будет рассчитан на основе старого токена. Очевидно, что такой запрос не будет авторизован сервисом Бизнес.Ру, так как он будет рассчитывать пароль с использованием нового токена, при этом сервис вернет приложению HTTP-ответ с кодом статуса 401 Unauthorized. Получив такой ответ приложение должно послать сервису Бизнес.Ру GET-запрос вида:
https://myaccount.business.ru/api/rest/repair.json?
app_id=app_id_value&app_psw=app_psw_value
Вместо имени модели в данном запросе фигурирует ключевое слово repair (восстановление). app_id_value - сохраненное в базе данных приложения значение идентификатора интеграции, app_psw_value - значение пароля, которое в данном случае рассчитывается по упрощенной формуле (без учета токена):
app_psw = MD5(secret + params_string),
где параметром функции MD5 служит конкатенация компонентов:

secret - предоставленный приложению секретный ключ;
params_string - url-кодированная строка единственного параметра запроса - app_id.
Получив запрос восстановления токена, сервис Бизнес.Ру авторизует его также по упрощенной схеме, и, в случае успеха отправляет приложению ответ, результирующий массив которого содержит новый токен и рассчитанный на его основе пароль app_psw (также без учета старого токена).
Получив HTTP-ответ с кодом статуса 200 Ok, приложение осуществляет проверку результата - преобразует JSON-строку результата в массив, проверяет наличие элементов массива token и app_psw, создает копию массива без app_psw, на основе которой получает JSON-строку и вычисляет MD5(secret + result_json). Если полученное значение совпадает со значением элемента массива app_psw, считается, что проверка пройдена. Токен восстановлен, последующие запросы формируются и проверяются по полной схеме (с учетом токенов).

Библиотека business_ru_api_lib.php

Если разработчик использует для написания интеграции язык PHP, для него будет полезна библиотека business_ru_api_lib.php, Библиотека состоит из единственного класса Business_ru_api_lib, инкапсулирующего всю логику по формированию/обработке паролей запросов/ответов, а также их отправке/приему по сети. Взаимодействие с сетью происходит с помощью функций libcurl. Библиотека cbusiness_ru_api_lib.php может быть также использована программистами других языков в качестве пособия.
При получении библиотеки разработчиком интеграции, предоставленный приложению секретный ключ должен быть прописан в качестве значения приватной переменной класса Business_ru_api_lib $secret.
Перед использованием любого из методов класса Business_ru_api_lib приложение должно создать объект данного класса при помощи оператора new. Конструктор класса имеет вид:
__construct($app_id, $token, $address, $sleep_on_limit_reached = false)
Параметры:
$app_id - идентификатор интеграции,
$token - текущее значение токена,
$address - домен аккаунта Бизнес.Ру, на который должны слаться запросы,
$sleep_on_limit_reached - активизирует функцию sleep() при получении ответа с кодом 503.
Для отправки запроса к API сервиса Бизнес.Ру служит метод:
array request(string $action, string $model [, array $params]),

Параметры:
$action - одно из четырех предопределенных действий (‘get’, ’post’, ‘put’, ‘delete’),
model - модель данных, к которой адресуется запрос,
$params - параметры запроса.
Массив $params должен содержать только параметры, определяемые прикладной логикой. Системные параметры, служащие для авторизации запроса сервисом Бизнес.Ру (app_id и app_psw) присоединяются к запросу внутри функции.
Возвращаемое значение: в случае получения от сервиса Бизнес.Ру HTTP-ответа со статусом 200 Ok и его успешной проверки на стороне приложения, метод вернет результирующий массив, включающий в себя собственно результат запроса, а также новый токен. Результат запроса должен быть отправлен на обработку прикладной логике приложения, а новый токен записан в базу данных, для использования при формировании следующего запроса.
В данной ситуации поле status результирующего массива может иметь значение ‘ок’ либо ‘error’. Первый случай возникает, если запрос был успешно выполнен, второй - если запрос не может быть выполнен по причине, возникающей на уровне прикладной логики, не связанной с его авторизацией. Более подробно структура результирующего массива, а также классификация ошибочных ситуаций будет рассмотрена ниже.
Если запрос приложения не пройдет авторизацию на сервисе Бизнес.Ру, вернется HTTP-ответ со статусом 401 Unauthorized. В этом случае поле status результирующего массива будет иметь значение “error”, а поле error_code - “http:401”.
Если ответ сервиса Бизнес.Ру не пройдет проверку на стороне приложения, поле status результирующего массива будет иметь значение “error”, а поле error_code - “auth:1”.
В случае, если метод request возвращает ошибку приложение должно инициировать процедуру восстановления токена, если же это не устранит проблему, необходимо связаться с технической поддержкой скрвиса Бизнес.Ру.
Процедура восстановления токена реализована в методе:
array repair(),
Параметры: данный метод не имеет параметров.
Возвращаемое значение: в случае успеха, метод возвратит результирующий массив, поле status которого будет иметь значение “ok”, а поле token - новый токен, который приложение должно записать в свою базу данных для формирования следующего запроса.
Если поле статус результирующего массива будет иметь значение “error”, значит процедура восстановления токена не привела к желаемому результату. В этом случае следует связаться с технической поддержкой сервиса Бизнес.Ру.

При создании объекта класса Business_ru_api_lib с целью выполнения процедуры восстановления токена. в качестве значения параметра $token может передана пустая строка (токен потерян).
Авторизация запроса на удаление приложения реализована в методе bool deinstall().
Параметры: данный метод не имеет параметров.
Возвращаемое значение:
true - запрос авторизован, приложение должно удалить из своей базы данных соответствующую запись;
false - запрос не авторизован, приложение не выполняет никаких действий.

Запросы к API

В данном разделе будут рассмотрены параметры запросов и поля ответов имеющие отношение к манипуляциям с базой данных аккаунта Бизнес.Ру (прикладной логике). Системные элементы запросов и ответов, обеспечивающие авторизацию запросов и проверку ответов, были рассмотрены выше и далее упоминаться не будут.
Обязательным элементом каждого запроса является модель данных, она однозначно определяет таблицу, к которой адресован запрос. Параметры запроса уточняют выполняемое им действие.
Ответ всегда содержит поле status, которое может принимать одно из двух значений: “ok” - запрос успешно выполнен, “error” - при выполнения запроса возникла ошибка.
В случае успеха, ответ будет также содержать поле result - собственно, результат выполнения запроса.
В случае ошибки, в ответе будут также присутствовать поля error_code, предназначенное для программной обработки ошибочной ситуации и error_text - ее описание для человека. Код ошибки имеет структуру source:number, где source - источник ошибки, number - номер ошибки. На уровне прикладной логики различаются два источника ошибок:
1. “common” - ошибки этого типа могут возникать в любых моделях данных, они определяются реализацией хранилища информации - реляционной СУБД. Примерами таких ошибок могут служить попытка присвоения полю таблицы значения несоответствующего типа, либо попытка присвоения внешнему ключу таблицы несуществующего значения.
2. “model” - ошибки, специфичные для конкретной модели, либо для ряда однотипных моделей. К таким ошибкам могут приводить действия, формально допустимые с точки

зрения структуры базы данных, однако противоречащие ограничениям, вытекающим из предметной области. Например: попытка присвоения документу договора, заключенного между организацией и контрагентом, несоответствующим организации и контрагенту документа.
На приложение накладываются ограничение в 500 запросов к API одного аккаунта за 5 минут. Время рассчитывается с момента первого запроса в серии.
При превышении лимита, API становится недоступным до окончания текущих 5 минут. В таком случае код ответа сервера - 503.
Лимит 500 запросов за 5 минут может быть увеличен. Для этого необходимо приобрести пакет с необходимым количеством запросов. Это можно сделать внутри системы в разделе "Настройки аккаунта\Выбор тарифного плана", тарифная опция "Запросы к API".

GET-запрос

GET-запрос служит для постраничного чтения информации из заданной модели данных. Поле result ответа представляет запрошенную страницу модели - массив с числовыми индексами, каждый элемент которого - запись соответствующей таблицы (ассоциативный массив, где ключи - имена полей таблицы).
Если запрос не содержит параметров, в ответе будет возвращена первая страница запрошенной модели данных. Размер страницы по умолчанию - 250 записей. Он может быть изменен в меньшую сторону путем указания параметра limit. Указав параметр page, приложение может получить любую необходимую страницу модели.

GET-запрос: Получение помощи по модели

Указав в запросе параметр “help=1” можно получить описание модели данных, В этом случае поле result ответа будет содержать следующую информацию:

- список запросов из числа get, post, put, delete, применимых к данной модели;
- список всех полей модели с исчерпывающей информацией о каждом поле, включая:
  - наименование поля (индекс ассоциативного массива), служит для идентификации поля в GET-, POST- и PUT-запросах;
  - текстовое описание поля в терминах предметной области;
  - тип данных поля из числа string, int, float, bool, date, color;
  - список запросов, присутствие в которых этого поля обязательно;
  - если поле является внешним ключом, указывается имя модели, на которую оно ссылается;
  - значение по умолчанию для поля. Данная информация, как правило, относится к запросу POST. Если это предусмотрено прикладной логикой, значение по

умолчанию присваивается полю, если оно отсутствует в запросе. Для некоторых полей значение по умолчанию приводится явно, для других в описании приводится фраза “Значение по умолчанию рассчитывается системой”. Например, это справедливо для полей number (номер документа) и date (дата документа), присутствующих практически во всех моделях, представляющих учетные и финансовые документы. Эти поля могут быть проинициализированы пользователем явно, в противном случае их присвоит система. В общем случае “Значение по умолчанию рассчитывается системой” применимо к тем полям, значения которых предзаполняются системой и при работе в пользовательском интерфейсе аккаунта Бизнес.Ру;
    - в некоторых моделях имеются поля изменение которых запросом PUT не допускается. В этом случает соответствующая пометка также присутствует в описании.
- список дополнительных полей модели с описанием. Список доп. полей представлен двумя блоками. Блок additional_fields содержит человекопонятное описание доп. полей, в виде массива: Уникальный идентификатор доп. поля => Текстовое описание доп. поля. Блок additional_fields_extended содержит структурированное описание, более подходящее для программной обработки, в виде массива: Уникальный идентификатор доп. поля => Структурированное описание доп. поля. Уникальный идентификатор доп. поля служит для идентификации поля в GET-, POST- и PUT-запросах. Описание доп. поля включает наименование (add_field_name) и тип (add_field_type). В блоке additional_fields_extended для доп. полей типа Список содержится также список допустимых значений поля (add_field_values).
- числовой идентификатор дополнительного поля (индекс ассоциативного массива), служит для идентификации поля в GET-, POST- и PUT-запросах;
- наименование дополнительного поля, присвоенное ему при создании;
- тип дополнительного поля, присвоенный ему при создании;
- список ошибок типа common, возникновение которых возможно при работе с моделью;
- список ошибок типа model, возникновение которых возможно при работе с моделью.

Описания моделей данных, полученные с помощью GET-запросов (путем указания параметра help) служат дополнением к данному руководству разработчика. Работу с каждой моделью рекомендуется начинать с отправки к ней именно этого запроса, с последующим анализом полученной в ответ информации.

GET-запрос: Фильтрация данных

В качестве дополнительный параметров GET-запроса, могут быть использованы условия, накладываемые на значения полей модели данных. Данные условия будут переданы в блок WHERE sql-запроса, осуществляющего выборку записей из соответствующей таблицы.
Допускается использование следующих типов условий: простое сравнение, перечисление, диапазон, сопоставление с шаблоном.
Простое сравнение имеет вид: ‘field’ => ‘value’, преобразуется в sql-конструкцию field = value, применимо для полей всех типов.
Перечисление имеет вид: ‘field’ => [‘value1’, ‘value2’, ‘value3’, …], преобразуется в sql-конструкцию field IN(value1, value2, value3, …), применимо для полей типов string, int, float, date.
Диапазон имеет вид: ‘field’ => [‘from’ => ’from_value’, ‘to’ => ’to_value’], преобразуется в sql-конструкцию field >= from_value AND field <= to_value, применимо для полей типов float, date, datetime. Для указания границ диапазона можно использовать как оба значения (from, to), так и только одно из них.
Сопоставление с шаблоном имеет вид: 'field' => ['like' => 'pattern'], либо 'field' => ['ilike' => 'pattern'], преобразуется в sql-конструкцию field LIKE pattern, либо field ILIKE pattern, применимо для полей типа string. Согласно правилам sql, pattern представляет собой любое символьное выражение, с подстановочными символами '%' (ноль, один или несколько произвольных символов) и '_' (ровно один произвольный символ). Команда like обеспечивает чувствительный к регистру поиск. Команда ilike обеспечивает нечувствительный к регистру поиск.
С помощью условий можно фильтровать запрашиваемую информацию по нужным признакам. Например, получить запись с определенным идентификатором (сравнение), либо получить документы, статус которых принадлежит некоторому множеству (перечисление), либо получить документы, дата создания которых находится в некотором диапазоне (диапазон), либо получать товары, наименования которых содержат заданную подстроку (сопоставление с шаблоном).
Указывая диапазон поля updated можно запросить лишь те записи модели, которые были изменены в течение заданного периода времени. Используя данную возможность, можно, построить логику, обеспечивающую считывание только тех записей, которые были добавлены или изменены в период с предыдущего считывания, что обеспечит приложению синхронизацию своей базы данных с базой данных сервиса Бизнес.Ру.

GET-запрос: Сортировка данных

Параметр order_by служит для указания сортировки запрашиваемой выборки и преобразуется в одноименную sql-конструкцию. Параметр имеет вид: ‘order_by’ => [‘field1’ => sort_type, ‘field2’ => sort_type, …], где sort_type - одно из двух значений: ‘ASC’ (сортировка в порядке возрастания), либо ‘DESC’ (сортировка в порядке убывания). Допускается сокращенная форма записи: ‘order_by’ => ‘field’, которую удобно использовать для указания простой сортировки по единственному полю в порядке возрастания. Для всех остальных случаев следует использовать полную форму записи.

GET-запрос: Получение данных из связанных моделей

При запросах к моделям данных, реализующим списки документов и имеющим содержимое в виде перечня товаров и услуг, может использоваться дополнительный параметр "with_goods=1". Данный параметр позволяет получить документ с перечнем ассоциированных с ним товаров и услуг одним запросм.
Дополнительный параметр "with_goods=1" может быть применен к следующим моделям:

- offers
- customerorders
- customerinvoices
- realizations
- reservations
- returnbuyers
- supplierorders
- supplierinvoices
- supplies
- returnsuppliers
- remains
- charges
- postings
- shiftings
- inventories
- internalorders
- deals

При запросах к моделям данных, имеющим ссылки на другие модели, может использоваться дополнительный параметр "extended=1". В этом случае в ответе на запрос, помимо идентификаторов записей связанных моделей, будет содержатся

основная информация соответствующих записей связанных моделей. Дополнительный параметр "extended=1" может быть применен к следующим моделям:

- customerorders
- customerordergoods
- realizations
- realizationgoods
- goods
- partners
- paymentout

GET-запрос: Получение количества записей в модели

При указании в GET-запросе к любой модели параметра count_only =>1, в ответе будет возвращено единственное поле count, отражающее общее количество записей в соответствующей модели. Если наряду с параметром count_only в запросе были указаны другие параметры, накладывающие ограничения на выборку, то поле count отразит количество записей в модели, удовлетворяющих заданным условиям.

POST-запрос

POST-запрос служит для добавления в модель данных новой записи. В параметрах запроса указываются значения, которыми должны быть проинициализированы поля новой записи. При получении запроса сервис Бизнес.Ру осуществляет валидацию полученной информации, если требуется, инициализирует какие-то из полей значениями по умолчанию, и добавляет запись в таблицу, В случае успеха, поле result ответа будет содержать идентификатор вновь добавленной записи, а также время ее добавления.

PUT-запрос

PUT-запрос служит для редактирования уже существующей в модели данных записи. Обязательным параметром запроса является идентификатор записи, которую требуется редактировать. Остальные параметры отражают новые значения полей записи. В случае успеха, поле result ответа будет содержать идентификатор отредактированной записи, а также время ее обновления.

DELETE-запрос

DELETE-запрос служит для удаления записи из модели данных . Обязательным

параметром запроса является идентификатор записи, которую требуется удалить. В случае успеха, поле result ответа будет содержать идентификатор удаленной записи, а также время ее удаления.

Работа с изображениями

С записями некоторых моделей данных могут быть ассоциированы галереи изображений. Например каждому товару может соответствовать множество его фотографий.
Информация об изображениях, ассоциированных с записями некоторой модели представлена полем images данной модели. Поле images представляет собой массив, каждый элемент которого содержит информацию об одном изображении и имеет вид:
array(2) {
"name" = "Имя файла с расширением",
"url" = "URL для скачивания файла"
}
Поле images может присутствовать в ответе на GET-запрос (получение текущих изображений), а также в качестве параметра POST- и PUT-звпросов. При включении параметра images в POST-запрос, с вновь созданной записью модели будут ассоциированы изображения, загруженные сервисом Бизнес.Ру с ссылок, перечисленных в массиве images. При наличии параметра images в PUT-запросе, текущие изображения соответствующей записи будут перезаписаны перечисленными в массиве images. Указание в параметре images PUT-запроса пустого массива приведет к удалению текущих изображений соответствующей записи.

Работа с дополнительными полями

Сервис Бизнес.Ру позволяет определять пользовательские свойства для некоторых своих сущностей. Это обеспечивается с помощью механизма дополнительных полей. При работе в аккаунта Бизнес.Ру, для сущностей, где это предусмотрено, пользователь может создавать дополнительные поля, указывая их наименование и тип, тем самым расширяя существующую структуру соответствующих сущностей. В дальнейшем дополнительные поля становятся доступными для манипуляций наряду с предопределенными полями.
API Бизнес.Ру также обеспечивает доступ к дополнительным полям. Список дополнительных полей, определенных для заданной модели данных можно получить с помощью GET-запроса с параметром “help=1”.
Для того чтобы получить информацию о текущих значениях дополнительных полей

модели данных, в GET-запросе к ней следует указать параметр "with_additional_fields=1". Если в данной модели предусмотрены дополнительные поля и они фактически в ней присутствуют, к массивам записей запрошенной модели будут добавлены элементы, каждый из которых имеет вид:
[идентификатор]=>значение,
где:

- идентификатор - числовой идентификатор дополнительного поля в базе данных аккаунта Бизнес.Ру;
- значение - значение дополнительного поля для текущей записи.

Наличие данной конструкции в POST- и PUT-запросах обеспечивает инициализацию дополнительных полей при добавлении новых записей и редактировании существующих.

Примеры запросов к API

Восстановление токена

```
// подключаем библиотеку
include 'business_ru_api_lib.php';

// создаем экземпляр класса Business_ru_api_lib
$api = new Business_ru_api_lib( $app_id, “”, $address );

// вызываем метод repair
$response = $api->repair();

// извлекаем токен
$token = $response[“token”];
// сохраняем токен для последующего использования
```

Отправка GET-запроса

```
// подключаем библиотеку
include “business_ru_api_lib.php”;

// создаем экземпляр класса
$api = new Business_ru_api_lib( $app_id, $token, $address );
```

```
// запрашиваем интерактивную помощь по модели goods
$response = $api->request( “get”,”goods”, [ “help” => 1 ] );

// либо запрашиваем полный список товаров и услуг
$response = $api->request( “get”,”goods” );

// либо запрашиваем полный список товаров, отсортированный в порядке возрастания идентификатора
$response = $api->request( “get”,”goods”, [“type” => 1, “order_by” => ”id”] );

 // либо запрашиваем полный список товаров, отсортированный в порядке убывания идентификатора
$response = $api->request( “get”,”goods”, [“type” => 1, “order_by” => [”id” =>’DESC’]] );

// либо запрашиваем полный список товаров и услуг с составной сортировкой
$response = $api->request( “get”,”goods”, [“order_by” => [”partner_id” =>’ASC’, ”full_name” =>’ASC’]] );

// либо запрашиваем список услуг, помещенных в архив
$response = $api->request( “get”,”goods”, [ “archive” => true, “type” => 2  ] );

// либо запрашиваем заказ покупателя с определенным идентификатором
$response = $api->request( “get”,”customerorders”, [ “id” => 123 ] );

 // либо запрашиваем заказы покупателя с заданными статусами
$response = $api->request( “get”,”customerorders”, [ “status_id” => [ 10, 21, 22, 23 ]  ] );

// либо запрашиваем заказы покупателя, дата создания которых находится в заданном диапазоне
$response = $api->request( “get”,”customerorders”, [ “date” => [ “from”=>“01.12.2016”, “to”=>“31.12.2016”  ]  ] );

// либо запрашиваем сделки, информация в которых была обновлена после заданного момента времени
$response = $api->request( “get”,”deals”, [ “updated” => [ “from”=>“01.12.2016 12:00:00”  ]  ] );
```

```
// извлекаем статус и токен
$status = $response[“status”];
$token = $response[“token”];
if( $status == “ok” ) { // успех
$result = $response[“result”]; // извлекаем данные
}
else if( $status == “error” ) { // ошибка

// извлекаем информацию об ошибке
$error_code = $response[“error_code”];
$error_text = $response[“error_text”];
}

// сохраняем токен для последующего использования
```

Отправка POST-запроса

```
// подключаем библиотеку
include “business_ru_api_lib.php”;

// создаем экземпляр класса
$api = new Business_ru_api_lib( $app_id, $token, $address );

// создаем товар с наименованием “Мороженое”
$response = $api->request( “post”,”goods”, [ “name” => “Мороженое” ] );

// извлекаем статус и токен
$status = $response[“status”];
$token = $response[“token”];
if( $status == “ok” ) { // успех
// извлекаем данные (идентификатор созданного товара и время создания)
$result = $response[“result”];
}
else if( $status == “error” ) { // ошибка
// извлекаем информацию об ошибке
$error_code = $response[“error_code”];
$error_text = $response[“error_text”]; }
```

```
// сохраняем токен для последующего использования
```

Обработка заказов покупателей

```
include 'business_ru_api_lib.php'; // подключаем библиотеку
$secret  = "Hgl6Q1GF8lwHZbRX3nq7fO8MEytNsdJ0"; // секретный ключ
$app_id  = "461979"; // Id интеграции
$address = 'https://myaccount.business.ru'; // адрес
$action  = 'get'; // операция
$model   = 'offers'; // модель
$date_to_compare_with = "01.01.2017 00:00:00"; // метка времени
$api = new Business_ru_api_lib( $app_id, '', $address ); // создаем экземпляр класса
Business_ru_api_lib
$api->setSecret( $secret ); // устанавливаем секретный ключ
$res = $api->repair(); // восстановление токена
$i = 1;
do {
   $token = $res[ 'token' ]; // извлечение токена из ответа сервера
   $api->setToken( $token ); // установка текущего токена
   $params = [ 'limit' => 250, 'page' => $i++ ];  // устанавливаем параметры пагинации
   $res = $api->request( $action, $model, $params ); // отправляем запрос
   foreach( $res[ 'result' ] as $item ) { // сканируем полученный массив заказов
покупателей
      if( strtotime( $item[ 'updated' ] ) > strtotime( $date_to_compare_with ) ) { // отбираем
заказы, обновленные позднее заданной метки времени
         ... //
         ... // выполняем необходимые действия с отобранными заказами
         ... //
         var_dump( $item [ 'updated' ] );
      }
   }
}
while( count( $res[ 'result' ] ) > 0 ); // делаем это до тех пор пока не извлечем все заказы
```

Работа с дополнительными полями

```
/***** Считываем данные модели partners с дополнительными полями *****/
include 'business_ru_api_lib.php'; // подключаем библиотеку
```

```
$secret  = "Hgl6Q1GF8lwHZbRX3nq7fO8MEytNsdJ0"; // секретный ключ
$app_id  = "461979"; // Id интеграции
$address = 'https://myaccount.business.ru'; // адрес
$action  = 'get'; // операция
$model   = 'partners'; // модель
$token = file_get_contents( 'business_ru_api_data.txt' ); // извлекаем токена из файла
$api = new Business_ru_api_lib( $app_id, $token, $address ); // создаем экземпляр класса Business_ru_api_lib
$params = [ 'with_additional_fields' => 1, ]; // включаем режим запроса дополнительных полей.
$res = $api->request( $action, $model, $params ); // отправляем запрос.
/***** добавляем контрагента, инициализируем доп. поле 23698467 значением "Особо ценный контрагент" *****/
/***** подразумевается что доп. поле имеет тип Text *****/
...
$action  = 'post'; // операция
...
$params = [
    "name"     => "Контрагент 1", // Имя контрагента
    "23698467" => "Особо ценный контрагент", // Дополнительное поле
];
$res = $api->request( $action, $model, $params ); // отправляем запрос.
/***** редактируем доп. поле контрагента *****/
...
$action  = 'put'; // операция
...
$params = [
    "id" => "23693351", // Идентификатор редактируемой записи
    "23698467" => "Просто ценный контрагент", // Дополнительное поле
];
$res = $api->request( $action, $model, $params ); // отправляем запрос.
```

Создание заказа покупателя

```
include 'business_ru_api_lib.php'; // подключаем библиотеку
$secret  = "Hgl6Q1GF8lwHZbRX3nq7fO8MEytNsdJ0";
$app_id  = "461979";
```

```
$address = 'https://myaccount.business.ru';
$api = new Business_ru_api_lib( $app_id, '', $address ); // создаем экземпляр класса Business_ru_api_lib
$api->setSecret( $secret ); // устанавливаем секретный ключ
$res = $api->repair(); // восстановление токена
//Создание нового заказа:
$response = $api->request['post', 'customersorders',['author_employee_id' => 12345, 'responsible_employee_id' => 12345, 'organization_id' => 12345, 'partner_id' => 12345]]
$customerOrderId = $response['result']['id']
//Добавление двух товаров к заказу:
$api->request['post','customerordergoods',['customer_order_id' => $customerOrderId, 'good_id' => 12345]]
$api->request['post','customerordergoods',['customer_order_id' => $customerOrderId, 'good_id' => 56789]]
```

Работа с удаленными записями

Записи некоторых моделей (товары, розничные точки, заказы ...) после удаления помещаются в корзину с возможность последующего доступа.
Для доступа к удаленным записям используется параметр deleted.
Пример:

```
//Выборка удаленных товаров(goods):
$api->request('get','goods',['deleted'=>1,'limit'=>10]);
```

Получение печатных форм документов

Сервис Бизнес.Ру позволяет получать по API печатные формы документов. Для получения списка шаблонов печатных форм, доступных для документов определенного типа, в GET-запросе к соответствующей модели данных следует указать параметр "get_templates => 1". В ответе придет список доступных шаблонов в формате "идентификатор шаблона" => "название шаблона".

Пример запроса:

```
$response = $api->request( “get”, ”offers”, [ "get_templates" => 1 ] );
```

Пример ответа:

```
"result" => [
    "1" => "Коммерческое предложение",
```

```
    "2" => "КП с изображением товаров"
]
```

Чтобы получить ссылку для скачивания печатной формы конкретного документа, следует выполнить GET-запрос к соответствующей модели с параметром "get_template_link => 1". В параметре id необходимо указать идентификатор целевого документа, а в параметре template_id - идентификатор шаблона из списка доступных шаблонов печатных форм, полученных на предыдущем шаге. Ссылка для скачивания печатной формы будет возвращена в поле href ответа. При указании в запросе параметра "with_sign => 1" будет возвращена печатная форма документа с подписью и печатью.

Пример запроса:

```
$response = $api->request( “get”, ”offers”, [
    "get_template_link" => 1,
    "id" => 301843,
    "template_id" => 1,
    "with_sign" => 1
] );
```

Пример ответа:

```
"result" => [
    "id" => 1752,
    "date" => "05.02.2020 12:17:20.352556 MSK",
    "name" => "Коммерческое предложение №1 от 05.02.2020 12:02.xlsx",
    "href" => "/file/get/?id=1752"
]
```

Функционал получения печатных форм доступен для моделей данных:

- offers
- customerorders
- customerinvoices
- realizations
- invoiceout
- returnbuyers
- supplierorders
- supplies
- returnsuppliers

- remains
- charges
- postings
- shiftings
- inventories
- cashin
- cashout
- paymentin
- paymentout
- retailchecks