ExaBooking

API для программ ExaApartments, ExaCamping, ExaHostel, ExaHotel, ExaRent.

Версия API от 28.12.2024

Содержание

О документе

Документ описывает программный интерфейс для взаимодействия через интернет с базой данных программ ExaApartments, ExaCamping, ExaHostel, ExaHotel, ExaRent. На данный момент API используется в модуле бронирования на сайте.

Основные понятия

Позиция — бронируемый объект. Для ExaApartments – это апартаменты, для ExaCamping – места, для ExaHostel – койко-места, для ExaHotel – номера (а также апартаменты и койко-места), для ExaRent – сдаваемые в аренду позиции.
Заказ — единица бронирования в системе. Заказ имеет уникальный номер, статус, период исполнения, общую стоимость, оплаченную сумму, может содержать несколько позиций.
Клиент — посетитель сайта, бронирующий позиции.

Обращение к базе данных

Обращение к базе данных реализовано через выполнение POST-запросов на определенный адрес.

Адрес запроса

ExaApartments: https://exadb.ru/cgi-bin/exaapartmentspublic.php
ExaCamping: https://exadb.ru/cgi-bin/exacampingpublic.php
ExaHostel: https://exadb.ru/cgi-bin/exahostelpublic.php
ExaHotel: https://exadb.ru/cgi-bin/exahotelpublic.php
ExaRent: https://exadb.ru/cgi-bin/exarentpublic.php

Параметры

Обязательные параметры при каждом запросе:

- db — идентификатор вашей базы данных (например, «htdemo» - для демонстрационной базы данных ExaHotel);
- command — имя команды (например, «CancelOrder»);

Остальные параметры зависят от команды и описаны ниже.
При передаче значений параметров необходимо выполнить следующие замены:

- Символ " (двойная кавычка) на {QUOTE};
- Символ \n (перевод строки) на {BR}.

Параметры типа Дата можно передавать в форматах: YYYY-MM-DD, DD-MM-YYYY, DD-MM-YY, YYYY.MM.DD, DD.MM.YYYY, DD.MM.YY, YYYY/MM/DD, DD/MM/YYYY, DD/MM/YY.
Параметры типа Логическое при значении Истина можно передавать в форматах (в любом регистре): 1, on, true, t, yes, y. Иначе значением считается Ложь.

Результат выполнения

После вызова запроса первым делом необходимо конвертировать полученный результат из кодировки Windows-1251 в кодировку UTF-8.
Возвращаемый результат состоит из двух частей: код результата и сами данные. Части разделены символом | (вертикальная черта).
Если запрос выполнен успешно, то код результата будет равен «OK». В этом случае можно использовать данные результата. В противном случае код результата содержит

описание/текст ошибки (например, «Неправильно задан период») или краткий идентификатор ошибки (например, «OBJECTNOTFOUND»).
Данные результата почти во всех случаях возвращаются в формате XML (исключение — когда для оптимизации необходима передача существенных объемов данных). Некоторые команды могут не возвращать данные.

Пример

Запрос:
https://exadb.ru/cgi-bin/exahotelpublic.php?db=htdemo&command=SetPeriod&date_from=2015-01-01&date_to=2015-01-31

Результат:
```
OK|<Result CartInfo="..."/>
```

Обратите внимание, хотя в примере запрос указан как GET-запрос, здесь и далее параметры следует подавать в формате POST-запроса.

Идентификаторы

Каждый объект (заказ, клиент и т.п.) в базе данных имеет свой уникальный идентификатор (ключ). Идентификатор выглядит как набор цифр и латинских символов. Например: «93C58A77EE2D4A49B4160F26E856BC1C» или «smlLifWeUm1s3nFx2DthBj».
Однако могут встретиться идентификаторы в виде числа (например, «115»), которые использовались в старых версиях программы.
Под идентификаторами позиций и идентификаторами услуг/товаров подразумеваются записи из таблиц в разделе «Бронирование на сайте», а не из основных таблиц позиций и услуг/товаров.

Доступ к командам

Большую часть команд можно вызывать свободно. Это те команды, которые используются в модуле бронирования. Например, получение списка свободных для бронирования позиций на период.
Некоторые команды доступны только пользователям программы. Например, получения списка сотрудников. Перед вызовом таких команд требуется один раз вызывать команду LoginUser с логином и паролем пользователя, чтобы получить идентификатор его сессии, а затем передавать эту сессию в команды.

Команды

BookingItemAddFile

Прикрепить файл к позиции.
Параметры:
item — идентификатор позиции.
filename — имя файла.

BookingItemDeleteFile

Удалить прикрепленный к позиции файл.
Параметры:
item — идентификатор позиции.
filename — имя файла без пути.

GetBookingItem

Получить информацию о позиции.
Параметры:
item — идентификатор позиции.
needvalues или skipvalues – объем возвращаемого результата. Подробнее смотрите в разделе «Параметры needvalues/skipvalues». В данном случае возможно использование значений: Styles, AttachedFiles, FileAttribs, DownloadFile, WebItem, WebItem.Price.
Результат:
Заполненный XML, содержащий информацию о позиции.

GetBookingItems

Получить информацию о всех позициях.
Параметры:
date_from, date_to — (необязательно) период существования (актуальности) позиций.
filial — (необязательно) идентификатор филиала (корпуса/зоны/пункта проката). Это дает возможность уточнить перечень выводимых позиций.
filialmark — (необязательно) идентификатор метки филиала (корпуса/зоны/пункта проката). Будут возвращены только те позиции, филиал которых имеет указанную метку.
itemtype — (необязательно) идентификатор категории позиции. Это дает возможность уточнить перечень выводимых позиций.
itemmark — (необязательно) идентификатор метки позиции. Будут возвращены только те позиции, которые имеют указанную метку.
needvalues или skipvalues – объем возвращаемого результата. Подробнее смотрите в разделе «Параметры needvalues/skipvalues». Значения аналогичны команде GetBookingItem.
Результат:
Заполненный XML, содержащий информацию о позициях.

GetBookingItemTypes

Получить перечень категорий позиций.
Параметры:
date_from, date_to — (необязательно) период существования (актуальности) категорий позиций.
Результат:
Заполненный XML, содержащий информацию о категориях.

GetAccessibleItems2

Используется для получения перечня свободных позиций на определенный период. Также возвращается стоимость бронирования каждой из свободных позиций на этот период.
Параметры:
date_from — дата начала периода. Значение должно быть позднее сегодняшнего дня.
date_to — дата окончания периода. Значение должно быть больше даты начала.
building — (необязательно) идентификатор филиала (корпуса/зоны/пункта проката). Это дает возможность уточнить перечень выводимых позиций. В этом значении также можно указать массив идентификаторов, разделенных запятыми.
filialmark — (необязательно) идентификатор метки филиала (корпуса/зоны/пункта проката). Будут возвращены только те позиции, филиал которых имеет указанную метку.
itemtypes — (необязательно) идентификатор категории позиции. Это дает возможность уточнить перечень выводимых позиций. В этом значении также можно указать массив идентификаторов, разделенных запятыми.
item — (необязательно) идентификатор позиции. Используется, если нужно проверить только конкретную позицию. В этом значении также можно указать массив идентификаторов, разделенных запятыми.
include_occupied — если этот флаг установлен, то в общем списке также возвращаются занятые позиции. У таких позиций выставляется флаг Occupied=”1”.
minprice — минимальная цена за сутки. Если указано, то возвращаются позиции с ограничением по этой цене.
maxprice — максимальная цена за сутки. Если указано, то возвращаются позиции с ограничением по этой цене.
sortmode — режим сортировки. Если задано «1», то возвращаемые позиции сортируются в случайном порядке.
Результат:
Заполненный XML, содержащий информацию по каждой доступной для бронирования позиции. Также содержится общая дополнительная информация по модулю бронирования: валюта, необходимость авторизации, текстовки и др.

GetEmployees

Получить список сотрудников.
Параметров нет.
Результат:
Список действующих сотрудников, по каждому возвращается идентификатор и ФИО.

GetFilials

Получить перечень филиалов/корпусов/зон.
Параметров нет.
Результат:
Список филиалов, по каждому возвращается идентификатор и наименование.

GetOccupyMatrix

Получение таблицы свободности/занятости позиций на каждый день на определенный период. Команда просматривает только те интернет-позиции, которые не являются категориями. Позиция считается занятой на дату, если в этот день невозможно арендовать позицию на сутки со стандартного времени аренды.
Параметры:
date_from — дата начала периода.
date_to — дата окончания периода. Значение должно быть больше даты начала.
Результат:

XML, содержащий информацию по позициям: по каждой из них выводятся даты занятости в формате yyyymmdd=1, например: 20190131=1 20190201=1. Не указанные даты считаются свободными.

CreateTemporaryOrder2

Создать заказ в базе данных. Заказ создается со статусом «Временный». Это позволяет забронировать позиции и дать клиенту время на оплату или подтверждение заказа, на ввод своих контактных данных. Если по истечению времени клиент так и не подтвердил/оплатил заказ, то заказ будет аннулирован или удален. Период времени (по умолчанию 3 часа) настраивается в системе.
Параметры:
date_from — дата начала периода. Значение должно быть позднее сегодняшнего дня.
date_to — дата окончания периода. Значение должно быть больше даты начала.
item_<идентификатор_позиции>=количество — заказанные позиции. Таких параметров можно передать несколько. Если вместо числа передано логическое значение, то подразумевается 1.
places — число бронируемых мест в номере (только для приложений ExaApartments и ExaHotel).
needprice — цена за сутки. Если значение указано, то в заказе будет включен ручной расчет с такой ценой за сутки (независимо от выбранных позиций). Если не указано, то автоматический расчет по тарифам, заданным в программе.
client_name — ФИО клиента.
client_phone — телефон клиента.
client_email — e-mail клиента.
checkexistingclientmode — режим проверки существования клиента: 0 (по умолчанию) – не проверять, всегда создавать клиента с такими данными и добавлять его в заказ, 1 – проверять уже существующего по переданному номеру телефона и при обнаружении добавлять его в заказ, 2 – проверять по адресу e-mail, 3 – проверять по ФИО (не рекомендуется, т.к. возможны полные тезки).
service_<идентификатор_услуги_товара>=количество — заказанные услуги/товары (необязательно). Таких параметров можно передать несколько. Если вместо числа передано логическое значение, то подразумевается 1.
promocode — промокод для получения скидки (необязательно).
notes — комментарий клиента к заказу (необязательно).
Результат:
Заполненный XML, содержащий информацию о созданном заказе: идентификатор заказа, стоимость и др.

ConfirmOrder2

Подтвердить временный заказ. При этом статус заказа меняется с «Временный» на «Необработанный».
Параметры:
online_client — идентификатор аккаунта клиента. Значение обязательно, если в настройках модуля бронирования установлено, что при создании заказа требуется регистрация. Этот идентификатор можно получить после того, как клиент ввел свои логин/пароль.
order — идентификатор заказа.
Результат:
Результат пустой либо сообщение об ошибке.

GetOrders

Получить перечень заказов.
Параметры:

temporary, unprocessed, reserved, active, finished, cancelled – флаги, заказы с какими статусами возвращать (соответственно: временный, необработанный, бронь, выполнение, завершено, аннулированный).
created_from, created_to – отобрать по периоду оформления заказа. Например, так можно получить заказы, созданные за последний день.
promouter – агент (идентификатор сотрудника, партнера или клиента), от которого пришел заказ.
customer – заказчик (идентификатор клиента).
execute_from, execute_to – период выполнения заказа.
sort – сортировка возвращаемого результата. Поддерживается параметр «executedatetime», что означает сортировка по дате/времени выполнения. Иначе сортировка по дате/времени оформления.
needvalues или skipvalues – объем возвращаемого результата. Перечень значений см. в команде ViewOrder.
Результат:
Перечень найденных заказов, которые соответствуют всем переданным условиям.
Информация по каждому заказу аналогична результату команды ViewOrder.

ViewOrder

Получить всю информацию о заказе.
Параметры:
id – идентификатор заказа.
needvalues или skipvalues – объем возвращаемого результата. Подробнее смотрите в разделе «Параметры needvalues/skipvalues». В данном случае возможно использование следующих значений:
Jobs – вывести состав заказа.
OrderAddits – получить дополнения к заказу (если включен модуль «Дополнения к заказу»).
FileCells – получить список прикрепленных рекомендуемых файлов.
AttachedFiles – получить список прикрепленных файлов (кроме рекомендуемых).
FileAttribs – (при FileCells или AttachedFiles) получить также атрибуты файлов: размер, дата создания и др.
DownloadFile – (при FileCells или AttachedFiles) получить также ссылки на скачивание файлов.
ConfirmOrPay – вывести информацию для кнопок подтверждения и/или предоплаты/оплаты заказа. При этом выводится информация для фискального чека.
Результат:
Информация о заказе: идентификатор заказа, номер заказа, период исполнения, дата и время оформления, заказчик, стоимость заказа, оплаченная сумма, комментарий и др.
Статус заказа (поле State) возвращается как перечисление со значениями:
6 – Временный заказ, клиент еще не подтвердил и не оплатил его. Возможно, заказ скоро будет удален системой в автоматическом режиме (зависит от настроек).
5 – Необработанный заказ (заказ уже подтвержден клиентом, но еще не рассмотрен администратором).
0 – Бронь.
1 – Выполнение.
4 – Завершен.
3 – Заказ аннулирован.

ViewOrderByKey

Получить всю информацию о заказе по номеру заказа.
Параметры:
key — 6-тизначный номер заказа.

Результат:
Команда возвращает результат, аналогичный ViewOrder.

OrderFailPay

Неуспешная попытка оплаты заказа. Возвращает адрес страницы, на которую следует перейти (этот адрес настраивается в модуле бронирования).
Параметры:
id — идентификатор заказа.

OrderSuccessPay

Успешная оплата заказа. Возвращает адрес страницы, на которую следует перейти (этот адрес настраивается в модуле бронирования).
Параметры:
id — идентификатор заказа.
bank_transaction_id — идентификатор транзакции в банковской системе. Это значение зависит от выбранного платежного сервиса и дается этим сервисом при успешной оплате.
summ — оплаченная сумма.

OrderAddCellFile

Прикрепление файла в ячейку рекомендуемых файлов для указанного заказа.
Параметры:
order – идентификатор заказа. Обязательный параметр.
index – индекс ячейки рекомендуемого файла. Нумерация с нуля. Если параметр не указан, то понимается как 0.
filename – имя файла. Обязательный параметр.
О способе загрузки файлов смотрите в разделе «Загрузка файлов на сервер».

OrderAddFile

Прикрепление файла для указанного заказа.
Параметры:
order – идентификатор заказа. Обязательный параметр.
filename – имя файла. Обязательный параметр.
О способе загрузки файлов смотрите в разделе «Загрузка файлов на сервер».

OrderDeleteCellFile

Удаление файла из ячейки рекомендуемых файлов указанного заказа.
Параметры:
order – идентификатор заказа. Обязательный параметр.
index – индекс ячейки рекомендуемого файла. Нумерация с нуля. Если параметр не указан, то понимается как 0.

OrderDeleteFile

Удаление прикрепленного файла из указанного заказа.
Параметры:
order – идентификатор заказа. Обязательный параметр.
filename – имя файла. Обязательный параметр.

AddAccount

Создание аккаунта для клиента, уже существующего в базе данных. Также см. команду RegisterClient.
Параметры:
client – идентификатор клиента. Обязательный параметр.

login – логин (обычно это адрес e-mail клиента). Обязательный параметр. Если аккаунт с таким логином уже существует в базе данных, то будет вызвана ошибка. Предварительно можно воспользоваться командой FindAccountByLogin.
newpassword1 – пароль. Обязательный параметр.
newpassword2 – подтверждение пароля, значение должно совпадать с newpassword1.
запятую).
Результат:
Возвращает данные как в команде Login.

LoginClient

Вход клиента.
Параметры:
email — логин, обязательное поле.
password — пароль.
Результат:
Возвращает: ФИО клиента, внутренний идентификатор сессии (Session), идентификатор записи личного кабинета (OnlineAccountID), телефон, e-mail и др.

RegisterClient

Регистрация нового клиента: создание записи о клиенте и создание аккаунта клиента. В базе данных поддерживается, что один клиент может иметь несколько аккаунтов.
Параметры:
name — ФИО клиента.
phone — телефон.
email — адрес электронной почты, обязательное поле. Является логином в аккаунте. В дальнейшем логин и адрес электронной почты клиента могут различаться.
birthday – дата рождения.
newpassword1 — пароль, обязательное поле.
newpassword2 — подтверждение пароля, значение должно совпадать с newpassword1.
ypostindex, ycountry, yregion, ycity, yaddress – адрес регистрации клиента (для организаций — юридический адрес). Соответственно: почтовый индекс, страна, регион (край, область), населенный пункт, уличный адрес (улица, дом, корпус, квартира/офис).
lpostindex, lcountry, lregion, lcity, laddress – адрес проживания клиента (для организаций — фактический адрес).
personaldataagreement – согласие на обработку персональных данных.
passport_number – номер документа, подтверждающего личность. Для паспорта это будет серия и номер через пробел, например, "1234 567890".
passport_givenby – кем выдан документ, подтверждающий личность.
passport_givenbycode – код подразделения (кем выдан документ).
passport_from – дата выдачи документа, подтверждающего личность.
nationality – гражданство.
snils – номер СНИЛС.
guardian – идентификатор клиента, который является родителем или другим законным представителем (указывается для ребенка).
iscompany – является ли клиент юр.лицом (организацией).
set_company_values_from_inn – выставить наименование, реквизиты и адрес организации по заданному значению ИНН. Для этого используется сервис dadata.
inn – ИНН (для организаций).
ogrn – ОГРН или ОГРНИП (для организаций).
kpp – КПП (для организаций).
okpo – ОКПО (для организаций).
include_marks – метки, которые следует включить (идентификаторы меток через запятую).

exclude_marks – метки, которые следует исключить (идентификаторы меток через запятую).
info1, info2, … – дополнительные пользовательские поля.
Результат:
Возвращает данные как в команде LoginClient.

LoginUser

Аутентификация сотрудника через свои логин-пароль в программе.
Параметры:
login – логин. Обязательный параметр.
password – пароль.
Результат:
Возвращает: идентификатор пользователя, идентификатор сотрудника, ФИО, внутренний идентификатор сессии, права на чтение и запись.

RestorePassword

Клиент запросил восстановление пароля.
Параметры:
email — адрес электронной почты, обязательное поле.

ChangePassword

Сменить пароль.
Параметры:
online_client_id — идентификатор записи личного кабинета, возвращаемый LoginClient или RegisterClient, обязательное поле.
curpassword — текущий пароль, обязательное поле.
newpassword1 — новый пароль, обязательное поле.
newpassword2 — подтверждение пароля, значение должно совпадать с newpassword1.

ViewClient

Получить всю информацию о клиенте.
Параметры:
id – идентификатор клиента.
needvalues или skipvalues – объем возвращаемого результата. Подробнее смотрите в разделе «Параметры needvalues/skipvalues». В данном случае возможно использование следующих значений:
PromoCodes – вывести список промокодов, принадлежащих клиенту.
LoyaltyCards – вывести список карт лояльности, принадлежащих клиенту.
Balance – вывести текущий баланс личной кассы клиента, а также историю пополнения баланса и списания средств с него.
Children – вывести также тех клиентов, для которых данный клиент является законным представителем (родителем).
FileCells – вывести перечень рекомендуемых к указанию файлов и какие из них прикреплены.
AttachedFiles – вывести список прикрепленных файлов.
DownloadFile – для файлов будет дополнительно возвращена ссылка для скачивания файла. Внимание: при этом создается временный файл, который доступен только небольшое время, достаточное для скачивания файла. Поэтому использовать DownloadFile нужно в момент нажатия пользователем на кнопку скачивания.
Результат:
Информация о клиенте: идентификатор клиента, ФИО, телефон, e-mail, дата рождения, паспортные данные, установленные метки и другое.

AddClient

Добавление клиента в базу данных.
Параметры:
name — ФИО клиента.
phone — телефон.
email — адрес электронной почты.
birthday – дата рождения.
ypostindex, ycountry, yregion, ycity, yaddress – адрес регистрации клиента (для организаций — юридический адрес). Соответственно: почтовый индекс, страна, регион (край, область), населенный пункт, уличный адрес (улица, дом, корпус, квартира/офис).
lpostindex, lcountry, lregion, lcity, laddress – адрес проживания клиента (для организаций — фактический адрес).
personaldataagreement – согласие на обработку персональных данных.
passport_number – номер документа, подтверждающего личность. Для паспорта это будет серия и номер через пробел, например, "1234 567890".
passport_givenby – кем выдан документ, подтверждающий личность.
passport_givenbycode – код подразделения (кем выдан документ).
passport_from – дата выдачи документа, подтверждающего личность.
nationality – гражданство.
snils – номер СНИЛС.
guardian – идентификатор клиента, который является родителем или другим законным представителем (указывается для ребенка).
iscompany – является ли клиент юр.лицом (организацией).
set_company_values_from_inn – выставить наименование, реквизиты и адрес организации по заданному значению ИНН. Для этого используется сервис dadata.
inn – ИНН (для организаций).
ogrn – ОГРН или ОГРНИП (для организаций).
kpp – КПП (для организаций).
okpo – ОКПО (для организаций).
include_marks – метки, которые следует включить (идентификаторы меток через запятую).
exclude_marks – метки, которые следует исключить (идентификаторы меток через запятую).
info1, info2, … – дополнительные пользовательские поля.

EditClient

Изменение данных о клиенте. Обязательным параметром является идентификатор.
Остальные параметры действуют следующим образом: если параметр передан (даже пустой), то он перезаписывает значение в клиенте, иначе никак не меняет.
Параметры:
id — идентификатор клиента.
Остальные параметры см. в команде AddClient.

FindAccountByLogin

Найти аккаунт по логину.
Параметры:
login — логин. По умолчанию, это значение параметра email в командах RegisterClient и LoginClient.
Результат:
Список найденных аккаунтов: пустой либо содержащий один узел. Если в базе дынных по каким-либо причинам заведено несколько акаунтов с одинаковым логином, то в списке будет более одного узла. Каждый узел содержит идентификатор акканта, идентификатор клиента.

FindClientByEmail

Найти клиента по адресу электронной почты. Поиск осуществляется не по части строки, а по полному совпадению.
Параметры:
email — адрес электронной почты, обязательное поле. Регистр написания неважен.
Результат:
Список найденных клиентов. Содержит перечень узлов, где каждый узел содержит результат, аналогичный команде ViewClient.

FindClientByPhone

Найти клиента по номеру телефона. Поиск осуществляется не по части строки, а по полному совпадению.
Параметры:
phone — номер телефона, обязательное поле. Ожидается в формате +70000000000, однако может содержать дополнительные символы.
Результат:
Список найденных клиентов. Содержит перечень узлов, где каждый узел содержит результат, аналогичный команде ViewClient.

ClientAddCellFile

Прикрепление файла в ячейку рекомендуемых файлов для указанного клиента.
Параметры:
client – идентификатор клиента. Обязательный параметр.
index – индекс ячейки рекомендуемого файла. Нумерация с нуля. Если параметр не указан, то понимается как 0.
filename – имя файла. Обязательный параметр.
О способе загрузки файлов смотрите в разделе «Загрузка файлов на сервер».

ClientAddFile

Прикрепление файла для указанного клиента.
Параметры:
client – идентификатор клиента. Обязательный параметр.
filename – имя файла. Обязательный параметр.
О способе загрузки файлов смотрите в разделе «Загрузка файлов на сервер».

ClientDeleteCellFile

Удаление файла из ячейки рекомендуемых файлов указанного клиента.
Параметры:
client – идентификатор клиента. Обязательный параметр.
index – индекс ячейки рекомендуемого файла. Нумерация с нуля. Если параметр не указан, то понимается как 0.

ClientDeleteFile

Удаление прикрепленного файла из указанного клиента.
Параметры:
client – идентификатор клиента. Обязательный параметр.
filename – имя файла. Обязательный параметр.

GetClientDownloadFile

Получение ссылки для скачивания файла клиента. После использования команды ViewClient с параметром needvalues=FileCells или needvalues=AttachedFiles можно узнать список файлов,

прикрепленных к клиенту. Для получения ссылки на скачивание конкретного файла используется команда GetClientDownloadFile.
Параметры:
client – идентификатор клиента. Обязательный параметр.
filename – имя файла. Обязательный параметр.

EditOrder

Изменение данных заказа. Обязательным параметром является идентификатор. Остальные параметры действуют следующим образом: если параметр передан (даже пустой), то он перезаписывает значение в заказе, иначе никак не меняет.
Параметры:
id — идентификатор заказа.
state — статус заказа. Число. Возможные варианты: 6 (временный), 5 (необработанный), 0 (бронь), 1 (выполнение), 4 (заказ завершен), 3 (заказ аннулирован), 2 (ремонт).
customer — идентификатор клиента.
promouter — идентификатор агента-партнера. Сюда следует подавать идентификатор не из таблицы агентов, а из таблицы партнеров.
summ — стоимость заказа. Если это значение передано и оно не совпадает с расчетом в заказе, то расчет заказа очищается, прайс переключается на «Ручной расчет», в расчет добавляется одна строка с указанной стоимостью.
notes — комментарий к заказу.
include_marks — метки, которые следует включить (идентификаторы меток через запятую).
exclude_marks — метки, которые следует исключить (идентификаторы меток через запятую).

Работа с заданиями

AddTask

Создать новое задание для сотрудника.
Параметры:
body – текст задания. Обязательный параметр.
executor – идентификатор сотрудника. Обязательный параметр.
targetdate – к какой дате необходимо выполнить задание.
targettime – к какому времени необходимо выполнить задание.
isimportant – важное.
Результат:
Результат аналогичен результату команды GetTask.

EditTask

Изменить задание.
Параметры:
task – идентификатор задания. Обязательный параметр.
body – текст задания.
executor – идентификатор сотрудника.
targetdate – к какой дате необходимо выполнить задание.
targettime – к какому времени необходимо выполнить задание.
isimportant – важное ли задание.
include_marks – метки, которые следует включить (идентификаторы меток через запятую).
exclude_marks – метки, которые следует исключить (идентификаторы меток через запятую).
Результат:
Результат аналогичен результату команды GetTask.

GetTask

Получить информацию по заданию.
Параметры:
task – идентификатор задания. Обязательный параметр.
needvalues или skipvalues – объем возвращаемого результата. Подробнее смотрите в разделе «Параметры needvalues/skipvalues». В данном случае возможно использование следующих значений:
Created – дата и время создания задания, а также какой сотрудник создал задание.
AttachedFiles – вывести список прикрепленных файлов.
DownloadFile – для файлов будет дополнительно возвращена ссылка для скачивания файла. Внимание: при этом создается временный файл, который доступен только небольшое время, достаточное для скачивания файла. Поэтому использовать DownloadFile нужно в момент нажатия пользователем на кнопку скачивания.
Результат:
По заданию возвращается: текст, сотрудник-исполнитель, важность, дата и время когда требуется выполнить и др.

GetTasks

Получить список заданий для выполнения сотрудниками.
Параметры:
executor – идентификатор сотрудника-исполнителя. Если не указано, то возвращаются все задания, доступные текущему сотруднику (по его правам доступа).
date_from, date_to – период выполнения.

skipactual – не возвращать актуальные (незавершенные) задания.
skipfinished – не возвращать завершенные задания.
Все параметры необязательные.
Результат:
По заданию возвращается идентификатор, а также все данные аналогично результату команды GetTask.

FinishTask

Пометить актуальное задание как завершенное.
Параметры:
task – идентификатор задания. Обязательный параметр.
finishnotes – замечания исполнителя.

ReturnTask

Вернуть завершенное задание в работу.
Параметры:
task – идентификатор задания. Обязательный параметр.

Работа с картами лояльности

EditLoyaltyCard

Изменить данные в карте лояльности.
Параметры:
loyaltycard – идентификатор карты лояльности. Обязательный параметр.
actual_from – срок начала действия карты. Если задать пустым, то понимается, что карта действовала всегда.
actual_to – срок окончания действия карты. Если задать пустым, то понимается, что карта будет действовать всегда.
owner – идентификатор клиента-владельца карты.
include_marks – метки, которые следует включить (идентификаторы меток через запятую).
exclude_marks – метки, которые следует исключить (идентификаторы меток через запятую).

FindFreeLoyaltyCard

Найти любую действующую карту лояльности указанной категории, которая не принадлежит ни одному клиенту. Команда используется для поиска свободной карты, чтобы затем привязать ее к клиенту командой EditLoyaltyCard.
Параметры:
type – идентификатор категории карт лояльности.
Результат:
Если найдено, то возвращаются данные по карте: идентификатор, срок действия, наименование и др.

Работа с промокодами

EditPromoCode

Изменить данные в промокоде. Доступно только изменение владельца.
Параметры:
promocode – идентификатор промокода. Обязательный параметр.
owner_client – идентификатор клиента-владельца карты.
owner_partner – идентификатор партнера-владельца карты.

GeneratePromoCode

Генерация промокода по заранее созданному шаблону. Команда используется для создания нового промокода, чтобы затем привязать его к клиенту или партнеру командой EditPromoCode.
Параметры:
template – идентификатор шаблона промокодов.
Результат:
Возвращаются данные по созданному промокоду: идентификатор, срок действия, наименование и др.

Параметры needvalues/skipvalues

Параметры needvalues и skipvalues используются в некоторых командах и влияют на объем возвращаемого результата, что, в свою очередь, влияет на объем обрабатываемых данных и скорость выполнения команды. Чем меньше в результате данных требуется, тем быстрее отработает команда.
Без использования этих параметров команды возвращают результат по умолчанию.
Можно использовать параметр needvalues либо skipvalues, чтобы указать, какие части результата нужны. Вместе сразу оба параметра указывать нельзя.
Если используется параметр needvalues, то программа вернет только то, что указано в его значении. Например:
needvalues=FieldInfo,Values
Если используется параметр skipvalues, то программа вернет то, что возвращает по умолчанию, за минусом того, что указано в значении этого параметра. Например:
skipvalues=Values
Значения внутри параметров needvalues и skipvalues указываются через запятую без пробелов, а их перечень зависит от конкретной команды. Например, при указании needvalues=AttachedFiles,DownloadFile для команды ViewOrder будет возвращен (дополнительно к обычной информации о заказе) список прикрепленных к заказу файлов, а также ссылки на скачивание этих файлов.

Загрузка файлов на сервер

Для некоторых команд (например, BookingItemAddFile) требуется передача пользовательских файлов на сервер. В зависимости от типа лицензионного ключа это сервер ExaOffice или сервер вашей организации, т. е. тот сервер, где в сети хранится центральная база данных.

Загрузка файла осуществляется в два шага. Также для этого потребуется создать отдельный php-файл на сайте.

Первый шаг. Отдельной операцией непосредственно загрузка файла на сервер, чтобы получить имя этого файла уже на сервере. Для этого нужно вызвать https://exadb.ru/phpcommands/exacodefileuploader.php с передачей туда следующих параметров:

1. back_url – адрес собственной страницы, которую необходимо вызвать. После успешной загрузки файла будет осуществлен переход на этот адрес с добавлением к нему конечного имени файла.
2. file – пользовательский файл.

Пример кода формы загрузки файла:

```
<form action="https://exadb.ru/phpcommands/exacodefileuploader.php?back_url=<?
echo
rawurlencode('https://mycompanywebsite.ru/mycode.php?param1=value1&param2=value2
&filename='); ?>" method="post" enctype="multipart/form-data" id="form1">
	<input type="file" name="file" id="file1">
</form>
```

Чтобы файл загружался сразу при выборе, можно добавить следующее:

```
<script>
	document.getElementById("file1").onchange = function() {
		document.getElementById("form1").submit();
	};
</script>
```

Второй шаг. Вызов нужной команды с указанием полученного имени файла.

Использование платежной системы

Программа интегрирована с несколькими платежными сервисами: ЮKassa (бывш. Яндекс.Касса), РобоКасса, LifePay, Сбербанк и др. Параметры интеграции настраиваются в самой программе.

Есть два способа вызова платежной формы для оплаты посетителем на сайте: стандартный вызов платежной ссылки ExaOffice, написание собственной платежной ссылки.

Платежная ссылка ExaOffice

Нужно вызвать следующую ссылку:

https://exacode.ru/exapayment/pay.php

В эту ссылку необходимо подать параметры, указанные ниже. Параметры можно передать как в виде GET-запроса, так и в виде POST-запроса.

Параметры:

3. PaymentService, YandexKassaShopId, YandexKassaScid, RoboKassaLogin, RoboKassaSignature, LifePayKey, Best2PayKey, SberbankApiUri, SberbankUserName, SberbankPassword — все эти данные возвращают командой ViewOrder. Набор параметров может зависеть от выбранного платежного сервиса, но лишние переданные параметры не приведут к ошибке.
4. MoneyToPayValue — сумма к оплате. Это значение также возвращается командой ViewOrder.
5. OrderDetails – описание заказа. Обычно эта строка формируется так: 'Заказ '.$xml['OrderKey'].'.
6. GlobalOrderID — глобальный идентификатор заказа в системах ExaOffice. Он содержит в себе название приложения, идентификатор базы данных и идентификатор заказа, разделенные двоеточием. Например: 'ExaApartments:apdemo:AE90972B2CC916A7'.

Собственная платежная ссылка

Написание платежной ссылки зависит от выбранного платежного сервиса. Ниже описан способ для эквайринга Сбербанка.

Ссылка на форму оплаты формируется на стороне Сбербанка (подробности необходимо смотреть в описании API Сбербанка «Инструкция по подключению интернет-магазина к платежному шлюзу»). При этом для ее формирования используются следующие параметры:

- SberbankApiUri, SberbankUserName, SberbankPassword, MoneyToPayValue, GlobalOrderID, OrderDetails (их получение описано выше).

Кратко, формирование и вызов ссылки выполняется таким образом:

1. Аутентификация клиента Сбербанка. В этот вызов подаются параметры:
    - Ссылка — взять значение из SberbankApiUri;
    - Пользователь API — взять значение из SberbankUserName;
    - Пароль пользователя API — взять значение из SberbankPassword.
2. Регистрация заказа в системе Сбербанка. В этот вызов подаются параметры:
    1. orderId — идентификатор заказа в ExaExcursions;
    2. orderAmount — взять значение из MoneyToPayValue и умножить на 100;
    3. ссылка в случае успешной оплаты;
    4. ссылка в случае ошибки оплаты;
    5. описание заказа;
3. При успешной регистрации заказа будет возвращена ссылка для перехода на платежную форму (параметр formUrl);

Для ссылки успешной оплаты можно использовать ссылку ExaOffice: https://exacode.ru/exapayment/sberbanksuccess.php?goid='.<GlobalOrderID>.'&summ='<MoneyToPayValue>

Либо сформировать собственный обработчик.

Для ссылки ошибки оплаты можно использовать ссылку ExaOffice: https://exacode.ru/exapayment/sberbankfail.php?goid='.<GlobalOrderID>

Либо сформировать собственный обработчик.

Рекомендуемая схема работы

Ниже описаны примерные шаги модуля бронирования, для которых разрабатывался настоящий API.

7. Вызов GetAccessibleItems2 — узнаем свободные позиции на выбранный пользователем период. Предлагаем выбор из этих позиций.
8. Ввод контактных данных клиента либо авторизация клиента.
9. Вызов CreateTemporaryOrder2 для создания временного заказа.
10. Подтверждение заказа — вызов ConfirmOrder2.

Примеры модуля бронирования

http://test.exaoffice.ru/exaapartments/
http://test.exaoffice.ru/exacamping/
http://test.exaoffice.ru/exahostel/
http://test.exaoffice.ru/exahotel/
http://test.exaoffice.ru/exarent/

Пример вызова

Пример получения перечня свободных позиций:

```
<?php

// общая часть

function CallExe($args) {
	foreach($args as $key => $val)
		$args[$key] = $key.'='.str_replace('"', '{QUOTE}', str_replace("\n", '{BR}', $val));
	$url = 'https://exadb.ru/cgi-bin/exahotelpublic.php';
	$result = file_get_contents($url, false, stream_context_create(array(
		'http' => array(
			'method'  => 'POST',
			'header'  => 'Content-type: application/x-www-form-urlencoded',
			'content' => http_build_query($args)
		)
	)));
	$result = iconv('windows-1251', 'utf-8', $result);
	$arr = explode('|', $result);
	return $arr;
}

function WriteExeErrorStr($message, $backbutton = true) {
	echo '<p style="color: red;">'.$message.'</p>';
	if ($backbutton)
		echo '<p align="center"><a href="#" onclick="history.back()"
class="major_button">Вернуться</a></p>';
}

function GetExeResultValue($exe_result) {
	return $exe_result[0];
}


function CheckExeResultValue($exe_result) {
	return (GetExeResultValue($exe_result) == 'OK');
}


function WriteExeError($exe_result, $backbutton = true) {
	switch(GetExeResultValue($exe_result)) {
		case 'INVALIDSESSION':
			header('Location: login.php');
```

```
                        exit;
                case 'UNKNOWNCOMMAND': WriteExeError('Команда в запросе не задана или задана
некорректно.', $backbutton); break;
                case 'INVALIDLOGIN': WriteExeError('Неверный пароль или пользователь с таким логином
не существует.', $backbutton); break;
                case 'OBJECTNOTFOUND': WriteExeError('Объект не указан или не существует в базе
данных.', $backbutton); break;
                default: {
                        if ($exe_result[0] == '')
                                WriteExeErrorStr('Ошибка при выполнении запроса.', $backbutton);
                        else
                                WriteExeErrorStr($exe_result[0], $backbutton);
                        break;
                }
        }
}

function CheckExeResult($exe_result, $backbutton = true) {
        if (CheckExeResultValue($exe_result)) return true;
        WriteExeError($exe_result, $backbutton);
        return false;
}

function GetResultValue($exe_result, $index) {
        return $exe_result[$index];
}

function GetXmlResult($exe_result) {
        $string = GetResultValue($exe_result, 1);
        return simplexml_load_string($string);
}

// получение позиций и вывод их простым списком

$args['db'] = 'htdemo'; // идентификатор базы данных
$args['command'] = 'GetAccessibleItems2'; // команда
$args['date_from'] = '2015-01-01';
$args['date_to'] = '2015-01-31';
$exe_result = CallExe($args);
if (CheckExeResult($exe_result)) {
        $xml = GetXmlResult($exe_result);
        $items = $xml->Items;
        foreach($items->Item as $item) {
                echo '<p>'.$item['Name'].'</p>';
        }
}

?>
```

Контакты

По всем вопросам обращайтесь: info@exaoffice.ru