ExaExcursions

Версия API – 3

Содержание

О документе

Документ описывает программный интерфейс для взаимодействия через интернет с базой данных программы ExaExcursions. На данный момент API используется в модуле бронирования экскурсий на сайте.

Основные понятия

Экскурсия — общее понятие посещения одного или нескольких экскурсионных объектов. Обычно, экскурсия сопровождается расписанием, прайсом. Также может иметь маршрут движения транспорта по умолчанию (используемый для сбора экскурсантов на точках посадки).
Рейс — выезд на конкретную экскурсию в заданные дату/время. Для рейса может назначаться водитель, транспорт. Также назначается конкретный маршрут движения транспорта, который может отличаться от маршрута в экскурсии по умолчанию.
Заказ — единица бронирования в системе на рейс. Заказ имеет стоимость, может содержать несколько клиентов/экскурсантов.
Клиент — человек, бронирующий экскурсию.
Экскурсант — человек, посещающий экскурсию. Разница между клиентов и экскурсантом в том, что забронировать экскурсию может один человек, но не для себя, а для других.
Маршрут — движение транспорта по точкам посадки для сбора экскурсантов.
Точка посадки — место сбора экскурсантов для посадки в транспорт. Каждая точка может иметь свой адрес в городе. Маршрут состоит из одной или нескольких точек посадки.

Обращение к базе данных

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

Адрес запроса

https://exadb.ru/cgi-bin/exaexcursionspublic.php

Параметры

Обязательные параметры при каждом запросе:

- db — идентификатор вашей базы данных (например, «exdemo» - для демонстрационной базы данных);
- command — имя команды (например, «CancelOrder»);
- version — должно равняться «3» для текущей версии API.

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/exaexcursionspublic.php?db=exdemo&command=CancelOrder&id=1440G5E2820

Результат:
```
OK|<Result><Order ID=”144AE90G” OrderKey=”89020011” TourDate=”2018.01.22” .../></Result>
```

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

Идентификаторы

Каждый объект (экскурсия, заказ и т.п.) в базе данных ExaExcursions 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 ExaExcursions.
Важно помнить один момент при работе с идентификаторами рейсов. Рейс создается автоматически при создании первого заказа на него. К примеру, если есть экскурсия в Кубинку и она имеет расписание по будням в 12:00. Посетитель сайта при бронировании увидит, что можно создать заказ заранее, например, за 1 месяц. Однако на этот момент рейса еще не существует в базе данных. Когда система выдавала расписание на будущее, она не вернула идентификатор рейса для этой даты. Как только посетитель создаст свой заказ, рейс будет реально создан и ему будет присвоен идентификатор. Такую особенность следует учитывать при кешировании данных о расписании.

Команды

GetOrderPrintPage3

Используется для печати документа для клиента. Это может быть ваучер на экскурсию или чек. HTML-шаблоны печати задаются в программе ExaExcursions. При вызове этой команды все макросы в шаблоне будут автоматически заменены на значения из переданного заказа.
Параметры:
id — идентификатор заказа.
type — поддерживаемые значения: orderpage (вызов печати со страницы заполнения списка экскурсантов), partner (вызов из кабинета агента).
index — номер шаблона (от 0 до 3).
user_verify — дата исполнения заказа (используется как дополнительная защита пользовательских данных от доступа извне).
При вызове из модуля бронирования экскурсии следует подавать пустые type и index.
Результат:
Заполненный HTML-документ, готовый для печати.

GetServices3

Вывести дополнительные услуги и товары, которые разрешены для заказа при бронировании экскурсии или для независимой продажи на сайте. Например, при бронировании экскурсии на шоколадную фабрику, клиент может дополнительно заказать 3 подарка на выбор.
Параметры:
mark_id — идентификатор метки. Если задано, то возвращаются только те услуги, у которых установлена данная метка.
tour_id — идентификатор экскурсии. Если задано, то возвращается перечень услуг, которые доступны при бронировании именно этой экскурсии.
execution_id — идентификатор рейса.
Результат:
Возвращает список услуг/товаров: их идентификаторы, наименования, единицы измерения, данные об их стоимости, набор меток по каждой услуге.

GetTourInfo

Получить всю информацию об экскурсии. Команда просто возвращает все данные, которые заданы в ExaExcursions на указанные рейс/экскурсию.
Параметры:
tour_id — идентификатор экскурсии.
execution_id — идентификатор рейса. Требуется для того, чтобы определить конкретные данные (на разных рейсах могут быть настроены разные цены, разные маршруты и др.). Если точность этой информации не важна, можно указать неконкретный рейс в формате yyyymmddhhnn- идентификатор_экскурсии. Например «201812311200-E2D4A49B4160F26E856B».
on_date — дата. Используется в том случае, когда необходимо просто получить информацию по туру на конкретный день (например, на сегодня). Рейс при этом не нужно указывать.
Результат:
Наименование экскурсии, описание/комментарий к стоимости, цены по каждому виду билета (взрослый, детский, без места, кат.2.взр., кат.2.дет.), точки посадки в маршруте и время прибытия к каждой точке, показывать ли выбор точки посадки, какие данные надо требовать с клиента (телефон, емейл, дата рождения, паспортные данные), нужно ли требовать данные со всех экскурсантов или только с одного.

Если передан параметр on_date, то дополнительно будет собрана информация по всем рейсам на указанный день. В результате в узлах MinPrices, MaxPrices, AvePrices будет возвращена информация по ценам (по каждому виду билета): минимальная, максимальная, средняя.

GetTourList3

Получить список экскурсий, у которых в системе настроено «Показывать на сайте».
Дополнительных параметров нет.
Результат:
Список экскурсий с идентификаторами и наименованиями.

GetTourTiming

Получить информацию о расписании экскурсии и числе свободных мест.
Параметры:
tour_id — идентификатор экскурсии.
date_from, date_to — запрашиваемый период. Если не указан, то подразумевается полгода с завтрашнего дня.
Результат:
В данном случае результат возвращается не в формате XML, а в простом формате CSV, где каждая строка содержит информацию о рейсе в виде:
дата_время_в_формате_yymmddhhnn;идентификатор_рейса;число_свободных_мест.
Например:

```
1801011000;93C58A77EE2D4A49B4160F26E856BC1C;22
1801011030;;999
1801011215;;999
1801011245;;50
```

Как было указано выше, для несуществующих еще рейсов идентификатор будет пустым. Если число мест возвращено равным 999, это означает, что еще нет ограничений на число мест (не назначен транспорт на рейс и нет других ограничений внутри экскурсии). В таком случае вместо числа мест нужно выводить строку вида «Еще есть много мест на эту экскурсию».

CreateOrder

Рассчитать стоимость заказа, создать новый заказ или обновить существующий заказ на экскурсию.
Расчет стоимости может использоваться для отображения (и динамического пересчета) стоимости заказа по мере ввода числа взрослых/детей на странице бронирования. При этом реально заказ в системе не создается.
Обновление заказа может потребоваться для изменения числа экскурсантов или других параметров в уже созданном заказе.
Заказы поддерживают статус «Временный». Можно создать заказ с таким статусом и дать клиенту время не торопясь вводить информацию по экскурсантам — на это время места на экскурсию будут забронированы. 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:
create — создавать ли заказ в системе.
online_client_id — идентификатор онлайн-клиента, если он указал свои логин/пароль.
tour_id — идентификатор экскурсии.
execution_id — идентификатор рейса. Если система вернула пустой идентификатор рейса (см. выше), то нужно подать значение в формате yyyymmddhhnn-идентификатор_экскурсии. Например, 1801011030-A49B4160F26E.
order_id — идентификатор существующего заказа, если идет обновление заказа.

outpoint_id — идентификатор точки посадки.
outpointaddress — уточнение адреса на точке посадки (если используется).
adult_count, children_count, free_count, noplace_count, cat2_count, cat2children_count — число экскурсантов.
client_count — по скольки экскурсантам введена информация (обычно 1).
client_name — ФИО первого экскурсанта.
client_phone — телефон первого экскурсанта.
client_email — e-mail первого экскурсанта.
client_birthday — дата рождения первого экскурсанта.
client_passport — номер паспорта первого экскурсанта.
Для остальных экскурсантов, если передается, то нужно указать номер экскурсанта после «client», например, client2_name — ФИО второго экскурсанта.
service_<идентификатор_услуги_товара>=количество — число заказанных услуг/товаров. Таких параметров можно передать несколько. Если вместо числа передано логическое значение, то подразумевается 1.
promocodes — список указанных промокодов через запятую.
notes — комментарий клиента к заказу.
Результат:
Если число экскурсантов превышает допустимое, то в коде результата будет возвращено соответствующее сообщение.
Если все хорошо, то возвращается результат аналогичный результату команды ViewOrder. В случае, если переданный параметр create не равен Истина, то некоторые данные в результате могут отсутствовать (например, идентификатор заказа).

ViewOrder

Получить всю информацию о заказе.
Параметры:
id — идентификатор заказа.
user_verify — дата исполнения заказа (используется как дополнительная защита пользовательских данных от доступа извне).
Результат:
Информация о заказе: идентификатор заказа, дата и время экскурсии, идентификаторы экскурсии и рейса, название экскурсии, число экскурсантов, стоимость заказа, оплаченная сумма, точка посадки, время прибытия транспорта на точку посадки, информация по клиентам, заказанные услуги и товары, таблица расчета, комментарий и др.
Тип заказа (поле Kind) возвращается как перечисление со значениями:
0 — Экскурсия
1 — Индивидуальный трансфер
2 — Групповой трансфер
3 — Индивидуальная экскурсия
4 — Аренда транспорта
5 — Приобретение услуг или товаров
Статус заказа (поле State) возвращается как перечисление со значениями:
0 — Актуальный заказ
3 — Заказ аннулирован
5 — Необработанный заказ (заказ уже подтвержден клиентом, но еще не рассмотрен диспетчером/администратором)
6 — Временный заказ, клиент еще не подтвердил и не оплатил его
8 — Невыход клиентов (заказ актуален, но экскурсанты не явились на экскурсию)

ViewOrderByKey

Получить всю информацию о заказе по номеру заказа.

Параметры:
key — 8-мизначный номер заказа.
user_verify — дата исполнения заказа (используется как дополнительная защита пользовательских данных от доступа извне).
Результат:
Команда возвращает результат, аналогичный ViewOrder.

ConfirmOrder

Подтвердить временный заказ. При этом статус заказа меняется с «Временный» на «Необработанный».
Параметры:
id — идентификатор заказа.
Результат:
Результат аналогичен результату команды 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.

LoginClient

Вход клиента.
Параметры:
email — логин, обязательное поле.
password — пароль.
Результат:
Возвращает: ФИО клиента, внутренний идентификатор сессии, идентификатор записи личного кабинета, 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.
passport_number — номер документа, подтверждающего личность.
passport_givenby — кем выдан документ, подтверждающий личность.

passport_givenbycode — код подразделения (кем выдан документ).
passport_from — дата выдачи документа, подтверждающего личность.
nationality — гражданство.
include_marks — метки, которые следует включить у клиента (идентификаторы меток через запятую).
Результат:
Возвращает данные как в команде LoginClient.

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.
Результат:
Информация о клиенте: идентификатор клиента, ФИО, телефон, e-mail, дата рождения, паспортные данные, установленные метки.

AddClient

Добавление клиента в базу данных.
Параметры:
name — ФИО клиента.
phone — телефон.
email — адрес электронной почты.
birthday – дата рождения.
personaldataagreement — согласие на обработку персональных данных.
passport_number — номер документа, подтверждающего личность.
passport_givenby — кем выдан документ, подтверждающий личность.
passport_givenbycode — код подразделения (кем выдан документ).
passport_from — дата выдачи документа, подтверждающего личность.
include_marks — метки, которые следует включить (идентификаторы меток через запятую).
exclude_marks — метки, которые следует исключить (идентификаторы меток через запятую).

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 RegisterClient.

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.

GetOrders3

Получить таблицу своих заказов. Используется для кабинета агента.

GetTours3

Получить список экскурсий в виде таблицы. Используется для кабинета агента.

SendSMSCode

Отправить SMS-код для подтверждения номера телефона, использованного для заказа экскурсии. Повторная отправка кода на один и тот же заказ возможна не ранее, чем через 60 секунд после предыдущей отправки. В системе должна быть настроена интеграция с p1sms.ru (сервис отправки SMS). Если интеграция не настроена, то в целях тестирования используется код подтверждения 1234.
Параметры:
id — идентификатор заказа.
phone — номер телефона. Это необязательный параметр. Если он не указан, то используется телефон первого экскурсанта (физ. лица).
Результат:
В случае успеха никакая дополнительная информация не возвращается. Иначе стандартным образом возвращается сообщение об ошибке: не найден заказ, прошло менее минуты после предыдущей отправки, нет номера телефона.

CheckSMSCode

Проверить SMS-код подтверждения номера телефона, использованного для заказа экскурсии.
Параметры:
id — идентификатор заказа.
code — код подтверждения.
Результат:
В случае успеха никакая дополнительная информация не возвращается. Иначе стандартным образом возвращается сообщение об ошибке: не найден заказ, код еще не отправлялся, неверный код и др.

GetServicePrice

Получить стоимость услуги/товара.
Параметры:
id — идентификатор услуги или товара.
Результат:
В параметре Price возвращается цена (например, «1500»). В параметре PriceStr возвращается цена в представлении для пользователя (например, «1 500 руб.»). Если параметры отсутствуют, значит цена в программе не задана.

BuyService

Оформить заказ на приобретение клиентом услуги или товара. Поддерживается заказ множества позиций одним заказом.
Параметры:
Service_идентификатор=количество — где: идентификатор — это идентификатор услуги или товара, количество — количество приобретаемых единиц (может быть логическим значением, тогда понимается как 1 при значении Истина, как 0 при значении Ложь). Можно указать любое число таких пар значений. Например:
Service_9C358645AED741F08FC811A854200FEB=2 Service_PpiLnEocce16Jlk31qQblr=yes.
ServiceDate_идентификатор=дата — необязательный параметр, используется в некоторых случах для уточнения даты исполнения услуги, например, если приобретается билет на мероприятие. Пример: ServiceDate_PpiLnEocce16Jlk31qQblr=2018.09.01.
order_date — дата исполнения заказа (необязательно). Если параметр не указан, то устанавливается сегодняшняя дата.
client_name — ФИО клиента.
client_phone — телефон клиента (необязательно).
client_email — e-mail клиента (необязательно).
В случае использования старого вида этой команды, когда поддерживался заказ единственной позиции, используются следующие параметры:
id — идентификатор услуги или товара.
count — количество приобретаемых единиц (должно быть больше нуля).
Результат:
Создается заказ на приобретение услуги/товара. Статус заказа — «Временный». Если по истечению времени (по умолчанию 3 часа) клиент так и не подтвердил/оплатил заказ, то заказ будет аннулирован или удален. Возвращаемый результат аналогичен результату команды ViewOrder.

Использование платежной системы

Программа ExaExcursions интегрирована с несколькими платежными сервисами: 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:

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

1. Необязательный шаг. Вызов GetTours3 – получаем список экскурсий, их названия и идентификаторы. Выводим список клиенту, чтобы он выбрал экскурсию.
2. Вызов GetTourTiming — узнаем расписание экскурсии и число свободных мест на каждое время. Выводим клиенту для выбора дня и времени.
3. Шаг ввода числа экскурсантов. Вызываем GetTourInfo для вывода цен и описания по стоимости. При вводе/изменении числа экскурсантов вызываем CreateOrder (без реального создания заказа) для расчета общей стоимости заказа.
4. Необязательный, но рекомендуемый шаг: дополнения к заказу. Вызываем GetServices3 для получения списка услуг и товаров, которые можно заказать сразу.
5. Если GetTourInfo возвращает, что для экскурсии необходимо выбрать точку посадки, то реализуем этот шаг. Выводим список точек, которые вернула команда GetTourInfo, для выбора клиентом, где он будет садиться в транспорт.

6. Ввод клиентом своих данных: ФИО, телефон, емейл и др., что настроено в системе и возвращает GetTourInfo.
7. Создание временного заказа — вызов CreateOrder со всеми собранными на предыдущих шагах значениями. Этот шаг можно поменять местами с предыдущим шагом. Временный заказ создан и места забронированы, клиент может перейти к оплате или подтвердить заказ без оплаты (как настроено в системе).
8. Подтверждение заказа — вызов ConfirmOrder. Либо оплата заказа — вызов OrderSuccessPay или OrderFailPay.

Результат GetTourInfo можно закешировать после первого вызова и помнить до смены экскурсии.

Пример модуля бронирования

http://test.exaoffice.ru/exaexcursions/booking3/

Пример вызова

Пример реализации запроса печати HTML-страницы по заказу:

```
<?php

function CallExe($args) {
	$args['version'] = '3';
	foreach($args as $key => $val)
		$args[$key] = $key.'='.str_replace('"', '{QUOTE}', str_replace("\n", '{BR}', $val));
	$url = 'https://exadb.ru/cgi-bin/exaexcursionspublic.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);
}

// пример выводит HTML-страницу для печати по конкретному заказу, используя шаблон, заданный в модуле бронирования в ExaExcursions
$args['db'] = 'exdemo'; // идентификатор базы данных
$args['command'] = 'GetOrderPrintPage3';
$args['id'] = $id; // идентификатор заказа из системы
$exe_result = CallExe($args);
if (CheckExeResult($exe_result)) {
	$xml = GetXmlResult($exe_result);
	$html = '';
	foreach ($xml->HTML as $htmlitem) {
		$html = $htmlitem['Text'];
	}
	echo $html;
}

?>
```

Контакты

По всем вопросам обращайтесь: info@exaoffice.ru