SmartBilet TicketSystem API v.2.5

Версия: 2.5.4.0

Дата: 19.11.2021

Оглавление

Изменения 5
Общее описание 7
Транспортный уровень 7
Общие сведения 7
Content-Type 7
Accept 7
Коды ответа HTTP 8
Успешное выполнение запроса 8
Ошибки 4XX (Client Errors) 8
Ошибки 5XX (Server Errors) 8
Сценарии взаимодействия 8
Получение сессионного ключа 8
Получение пользовательской сессии 8
Основной сценарий при бронировании билетов 9
Основной сценарий при бронировании билетов через шлюз 9
Команды протокола 10
Особенности протокола 10
Передача данных в API 10
Системные параметры запроса 10
Параметр $expand 10
Параметр $select 10
Параметр $orderby 10
Параметр $top 11
Параметр $skip 11
Параметр $inlinecount 11
Параметр $filter 11
Операторы OData для фильтрации 12
Функции OData для фильтрации 12
Получение сессии клиента API 13
Города 13
Билетные кассы 14
Категории мероприятий 14
Список категорий 14
Одна категория 15
Мероприятия 16

Список актуальных мероприятий 16
Одно мероприятие 17
Площадки 18
Список площадок 18
Одна площадка 19
Залы 19
Список залов 19
Сектора 20
Список секторов 20
Места 21
Список мест 21
Поставщики билетов 22
Список поставщиков билетов 22
События 22
Список событий 22
Одно событие 24
Свободные билеты 25
Промоакции 26
Корзина пользователя 27
Создание сессии пользователя 27
Поместить билет с местом в корзину 27
Поместить входной билет в корзину 28
Удалить билет из корзины 29
Удалить входной билет из корзины 29
Применить промоакцию к корзине 30
Удалить промоакцию из корзины 31
Просмотр корзины 31
Доступные способы доставки 33
Доступные способы оплаты 34
Брони пользователя 34
Создание брони из билетов в корзине 34
Список броней 36
Оплата брони 38
Отмена брони 39
Возврат билетов оплаченной брони 39
Добавление атрибута брони/билета 40

Отправка email о заказе покупателю ... 41
Получение файлов ... 41
Получение постера ... 41
Получение svg-файлов залов и секторов ... 41
Получение штрихкода ... 41
Получение PDF-файла заказа ... 42
Получение PDF-файла билета ... 42
Ошибки API ... 42
Формат ответа ... 42
Коды ошибок ... 43
Объекты API ... 44
Город (Town) ... 44
Билетная касса (TicketOffice) ... 44
Площадка (Venue) ... 44
Зал (VenueHall) ... 44
Сектор (Sector) ... 44
Место (Place) ... 45
Поставщик билетов (Provider) ... 45
Событие (Event) ... 45
Сектор события (EventSector) ... 46
Мероприятие (Action) ... 46
Категория (Category) ... 46
Тариф билета (TicketTariff) ... 47
Билет с местом (Ticket) ... 47
Билет в корзине (CartTicket) ... 47
Бронь (Order) ... 48
Доставка (Delivery) ... 48
Платёж (OrderPayment) ... 49
Информация о рекламной компании (UtmTagsInfo) ... 49
Описание (Description) ... 49
Корзина (Cart) ... 49
Промоакция (PromoAction) ... 50
Правило промоакции (PromoActionRule) ... 50
Атрибут (Attribute) ... 51
Информация о событиях в городе/площадке (EventsInfo) ... 51
Информация об интеграции с внешней системой (Integration) ... 51

Изменения

Версия	Дата	Список изменений
1.1	01.12.2014	Добавлены запросы на получение залов, секторов, мест. Добавлен запрос на отмену брони.
1.2	27.08.2015	Добавлен объект Provider – поставщик билетов. К событию добавлено поле ProviderId – идентификатор поставщика билетов
1.9.1	05.02.2016	Добавлен функционал промоакций
2.0.0	01.07.2016	Добавлен функционал сервисных сборов
2.2.0.1	22.09.2017	Исправлено правило применения промокода. Теперь идентификатор промоакции не требуется.
2.2.0.2	09.10.2017	Добавлено ограничение на отмену заказа. Теперь заказ на прошедшее событие нельзя отменить.
2.2.0.3	20.11.2017	Расширен функционал применения промоакции. Теперь поддерживается применение конкретного правила промоакции.
2.2.0.4	05.04.2018	В корзину добавлен идентификатор примененной промоакции.
2.2.0.5	28.05.2018	В корзину добавлен параметр «Оригинальная цена». В билеты корзины добавлены поля «идентификатор» и «объект» примененной промоакции. Теперь промоакцию билетов можно посмотреть командой *Cart('c88ed9b37caf4e259f993a95ed0e7f0f')?$expand=Tickets/PromoAction*
2.2.1.0	30.05.2018	У билетов в корзине и заказов появилось новое поле Attributes. Это позволит добавлять любую текстовую информацию о заказе – UTM-метки, внешний идентификатор, дополнительные контактные данные.
2.2.1.1	23.07.2018	Объёкт Event расширен полями: • MinCartPrice - Минимальная сумма билетов в корзине • MaxCartPrice - Максимальная сумма билетов в корзине • MinCartTickets - Минимальное количество билетов в корзине • MaxCartTickets - Максимальное количество билетов в корзине
2.2.2	19.03.2019	Добавлен метод получения PDF-файла билета.
2.2.3	25.07.2019	Добавлен метод отправки email-сообщения по заказу.
2.2.5	12.11.2019	Добавлена информация по рекламной компании при создании заказа.
2.3.6	01.02.2020	Добавлена информация по частичному возврату цены билета (193-ФЗ). Добавлен метод возврата билетов. Объёкт Event расширен полями: • ReturnTicket - Часть от цены билета в %, которая будет возвращена клиенту при возврате билетов.
2.3.7	02.08.2021	Добавлена возможность авторизации по HTTP-заголовку.
2.4.0	02.08.2021	Удалено поле DeliveryPrice у объекта Town. Добавлено поле KladrId у объекта Town. Добавлено поле EventsInfo у объекта Town. Добавлен метод запроса списка мероприятий. Добавлено поле IsPushkinsCard у объекта Action. Добавлен метод получения единого PDF-файла по заказу.

2.5.0	10.09.2021	Расширен метод оплаты заказа. Добавлена информация об оплате по банковской карте.
2.5.1	23.09.2021	Добавлен список интеграций у объекта Action. Добавлена возрастная категория 4+ у объекта Action.
2.5.2	28.10.2021	Добавлен список интеграций объекта Event. Добавлен список постеров у объекта Action.
2.5.3	01.11.2021	Добавлена возрастная категория 10+ у объекта Action.
2.5.4	19.11.2021	Добавлены списки ограничений у объектов Action и Event.

Общее описание

API предназначено для бронирования и/или продажи билетов на сайтах продаж сторонних агентов (партнеров), имеющих доступ в API. Запросы в API могут быть как с WEB-сервера, на котором располагается сайт продаж, так и с HTML-страницы посредством ajax-запросов.

Особенности:

1. В качестве транспортного протокола используется HTTPS
2. API построено на базе ASP.NET Web API OData v.3.
3. Ответы представляют собой сообщения в форматах:
    - Atom Pub (XML)
    - JSON “light”
    - JSON “verbose”
4. Протокол не использует пакетный режим. Один запрос – одна операция. В ответе приходит результат выполнения операции.
5. В протоколе используется аутентификация по клиентскому сертификату, HTTP-заголовку `certKey`, либо сессионному ключу, который передается в cookie или в параметре запроса `sessionKey`.

Транспортный уровень

Общие сведения

Все запросы передаются по протоколу HTTPS (HTTP over TLS) (RFC 2818) методами GET или POST. Сообщение протокола передается в теле POST-запроса в виде JSON строки, либо в URI GET-запроса. В API используются следующие способы аутентификации:

1. По клиентскому сертификату
2. По HTTP-заголовку (`certKey`) с данными о клиентском сертификате
3. По сессионному ключу (`sessionKey)`

В случае если не прошла аутентификация, возвращается ошибка 401 (`Unauthorized`). Некоторые запросы требуют обязательную аутентификацию по сертификату. При обращении к этим методам API по сессионному ключу, будет сгенерирована ошибка 403 (`Forbidden`).

Если в запросе не были указаны все необходимые данные, либо произошла ошибка бизнес-логики, то будет возвращена ошибка 400 (`BadRequest`), а в теле ответа будет описание ошибки.

При обработке запроса может возникнуть внутренняя ошибка сервера. В этом случае возвращается ошибка 500 (Internal server error), а в теле ответа содержится информация об ошибке сервера.

Content-Type

Все данные в запросах к api методами `POST` или `PUT`, необходимо передавать в виде json-строки, а в заголовке запроса `Content-Type` необходимо указывать `application/json`.

Ответы кодируются в кодировке UTF-8. Формат ответа определяется по заголовку запроса `Accept`.

Accept

В заголовке запроса Accept задается тип содержимого ответа, который сервер должен передать клиенту. Типы содержимого представлены в таблице ниже.

Content-Type	Описание
application/atom+xml	Ответ в формате Atom Pub (XML)

application/json	Ответ в формате JSON “light”
application/json;odata=verbose	Ответ в формате JSON “verbose”

Коды ответа HTTP

Успешное выполнение запроса

200 OK

Любой успешный GET, POST, PUT запрос в API будет содержать код ответа 200.

Ошибки 4XX (Client Errors)

404 Not Found

Формируется, когда HTTP-ресурс не найден, либо, когда не найден запрашиваемый объект.

400 Bad Request

Когда запрос клиента привел к какой-либо исключительной ситуации, ответ сервера будет содержать данный код, а также описание ошибки.

401 Unauthorized

Запросы в API требуют обязательной аутентификации, и если аутентификация не пройдена, сервер вернет пустой ответ, содержащий код 401.

403 Forbidden

Запросы на получение сессионного ключа и оплаты брони, требуют аутентификацию по сертификату, и если клиент выполняет запрос к такому методу с сессионным ключом (без клиентского сертификата), то сервер вернет 403 ошибку – доступ запрещен.

Ошибки 5XX (Server Errors)

500 Internal Server Error

Внутренняя ошибка сервера.

503 Service Unavailable

Сервис не доступен.

Сценарии взаимодействия

В этом разделе описаны сценарии взаимодействия с API. При описании взаимодействия используются следующие термины:

Клиент	Клиентское приложение (сайт), работающее с API
Пользователь	Пользователь клиентского приложения (сайта)
Сервер	Серверное приложение, обслуживающее запросы Клиента (API)

Получение сессионного ключа

Для того чтобы можно было работать с API не посредством клиентского сертификата, а с помощью аутентификации по сессионному ключу, необходимо выполнить запрос к API на получение сессионного ключа. После этого, можно отправлять запросы, например, со страницы напрямую, указав в URI запроса параметр `sessionKey`, либо установив cookie.

Получение пользовательской сессии

Запросы для работы с корзиной и с заказами требуют указание параметра `userSession` – сессия пользователя сайта. Каждый билет в корзине, а также бронь имеют ссылку на уникальный `userSession`, по которому можно будет определить корзину для конкретного пользователя сайта, а также вывести список броней пользователя.

Основной сценарий при бронировании билетов

1. Клиент получает сессионный ключ (в том случае если это требуется)
2. Клиент получает список событий с сервера и показывает его Пользователю
3. Пользователь выбирает событие и переходит к бронированию билетов
4. Клиент предлагает Пользователю выбрать места на схеме зала, либо из списка
5. Пользователь выбирает билет
    a. Клиент посылает запрос на добавление билета в корзину текущего Пользователя
    b. Сервер добавляет билет в корзину Пользователя с указанным `userSession` (при условии доступности этого билета для бронирования)
6. Пользователь переходит к своей корзине
    a. Клиент отправляет запрос на получение билетов в корзине текущего Пользователя
    b. Клиент отображает содержимое корзины Пользователя, кол-во билетов и общую стоимость брони с учетом сборов
7. Пользователь вводит данные о себе, выбирает способ доставки и получения билетов
8. Клиент отправляет на Сервер запрос на создание брони из билетов в корзине текущего Пользователя с указанными данными о броне
9. Клиент предлагает оплатить бронь
10. Пользователь переходит к оплате брони
11. Клиент переадресует пользователя на сайт-эквайринг
12. Сайт-эквайринг после успешной оплаты переадресует Пользователя на сайт Клиента
13. Клиент отправляет запрос об оплате брони на Сервер
14. Клиент получает информацию о броне, в которой может также содержаться штрихкод
    a. Клиент получает изображения штрихкодов
15. Клиент отправляет письмо пользователю с данными о броне, а также с штрихкодами электронных билетов, если таковые находились в броне.

Основной сценарий при бронировании билетов через шлюз

1. Клиент получает сессионный ключ (в том случае если это требуется)
2. Клиент загружает список событий с сервера
3. Клиент выбирает событие и запрашивает список свободных мест для бронирования
4. Клиент предлагает Пользователю выбрать места на схеме зала, либо из списка
5. Пользователь выбирает билет
    a. Клиент посылает запрос на добавление билета в корзину текущего Пользователя
    b. Сервер добавляет билет в корзину Пользователя с указанным `userSession` (при условии доступности этого билета для бронирования) и возвращает состояние корзины
6. Пользователь переходит к своей корзине
    a. Клиент отправляет запрос на получение билетов в корзине текущего Пользователя
    b. Клиент отправляет запрос на получение способов доставки (в том случае если это требуется)
    c. Клиент отображает содержимое корзины Пользователя, кол-во билетов и общую стоимость брони с учетом сборов, способы доставки
7. Пользователь вводит данные о себе, выбирает способ доставки и получения билетов
8. Клиент отправляет запрос на создание брони из билетов в корзине текущего Пользователя с указанными данными о броне на Сервер
9. Клиент предлагает оплатить бронь, отображая необходимую к оплате сумму из данных о броне
10. Пользователь переходит к оплате брони
11. Клиент отправляет запрос об оплате брони на Сервер
12. Клиент получает информацию о подтвержденной броне, в которой может также содержаться штрихкод

Команды протокола

Особенности протокола

API построен на базе ASP.NET Web API OData. Про OData (Open Data Protocol) версии 3.0 можно почитать по этой ссылке. Про особенности реализации протокола OData в ASP.NET Web API можно почитать тут.

Передача данных в API

Для передачи данных в API используется метод POST, в заголовке должен быть параметр `Content-Type: application/json`, в теле запроса должна быть JSON-строка.

Системные параметры запроса

Параметр $expand

По умолчанию все поля, имеющие комплексные типы (все типы, описанные в разделе Объекты ответов сервера), будут отсутствовать в выборке. Например, если запросить события, то у событий не будет полей Action, Venue, Sectors. Для того чтобы включить какое-либо поле в результат выборки, необходимо к URI запроса добавить параметр $expand, в котором через запятую можно передать все поля, которые необходимо «раскрыть» в ответе сервера. Например, такой запрос:

```
https://api.smart-bilet.ru/adm/api/Events?$expand=Action,Venue
```

вернет список событий с заполненными полями Action и Venue. При этом, данный параметр поддерживает вложенность. Например, у Action есть поле Category, и чтобы раскрыть и его, необходимо сформировать такой запрос:

```
https://api.smart-bilet.ru/adm/api/Events?$expand=Action/Category,Venue
```

Параметр $select

Для гибкого управление получаемыми данными из API, можно использовать параметр $select, в котором можно передать список полей, которые вы желаете видеть в ответе сервера. Например, если необходимо видеть только Id, дату начала события, наименование мероприятия и площадки, то достаточно сформировать следующий запрос:

```
https://api.smart-bilet.ru/adm/api/Events?$expand=Action,Venue&$select=Id,Date,
Action/Title,Venue/Title
```

Обратите внимание, что в запросе присутствует `$expand=Action,Venue`. API нужно указать, что в ответе нужно раскрыть эти поля, а уже потом указать, что в этих полях надо будет заполнить только Title.

Параметр $orderby

Параметр `$orderby` позволяет отсортировать ответ сервера по указанному порядку. Параметр содержит список свойств объекта, перечисленных через запятую, по которым необходимо выполнить сортировку. После наименования свойства должны следовать ключевые слова `asc` или `desc`, означающие соответвенно сортировку по возрастанию или по убыванию. Если `asc` или `desc` не были указаны, то API будет сортировать по возрастанию.

```
https://api.smart-bilet.ru/adm/api/Events?$orderby=Date desc,Action/Title asc
```

Запрос вернет список событий, отсортированных по дате события по убыванию и по наименованию мероприятия по возрастанию.

Параметр $top

Параметр `$top` указывает API, что в ответе нужны только первые n элементов. Значение параметра – положительное целое число.

```
https://api.smart-bilet.ru/adm/api/Events?$top=5
```

Запрос вернет 5 первых событий из списка событий. Если не указан параметр `$orderby`, API будет использовать сортировку по дате начала события.

Параметр $skip

Параметр `$skip` указывает API, что не надо включать в ответ первые n элементов ответа. Значение параметра – положительное целое число.

```
https://api.smart-bilet.ru/adm/api/Events?$skip=5
```

Запрос вернет события, начиная с 6-го элемента в списке событий.

Параметры `$top` и `$skip` могут использоваться вместе. При этом в независимости от положения параметров в URI запроса, сначала будет применяться параметр `$skip`, а затем `$top`.

```
https://api.smart-bilet.ru/adm/api/Events?$top=5&$skip=2
```

Запрос вернет события с 3-го по 7-е из списка событий (пропускем 2 события, берем 5 событий)

Параметр $inlinecount

Параметр `$inlinecount`, со значением `allpages`, указывает API, что необходимо вернуть общее кол-во элементов в ответе сервера. Пример:

```
https://api.smart-bilet.ru/adm/api/Events?$inlinecount=allpages
```

Параметр `$inlinecount` со значением `none` (так же, когда в запросе нет этого параметра), говорит API, что не надо возвращать общее кол-во элементов в ответе сервера. Этот параметр игнорирует значения параметров `$top`, `$skip` и `$expand` и всегда возвращает общее кол-во объектов в ответе.

Параметр $filter

Параметр запроса `$filter` позволяет отфильтровать выборку по любому полю.

```
https://api.smart-bilet.ru/adm/api/Events?$expand=Action&$filter=Action/CategoryId eq 6
```

Запрос вернет все события, мероприятие которых имеет id категории равную 6.

Также, в данном параметре запроса можно использовать функции.

```
https://api.smart-bilet.ru/adm/api/Events?$filter=year(Date) eq 2014 and month(Date) eq 7
```

Запрос вернет все события, которые проходят в июле 2014 года.

Операторы OData для фильтрации

Operator	Description	Example
Logical Operators		
eq	Equal	/Suppliers?$filter=Address/City eq 'Redmond'
ne	Not equal	/Suppliers?$filter=Address/City ne 'London'
gt	Greater than	/Products?$filter=Price gt 20
ge	Greater than or equal	/Products?$filter=Price ge 10
lt	Less than	/Products?$filter=Price lt 20
le	Less than or equal	/Products?$filter=Price le 100
and	Logical and	/Products?$filter=Price le 200 and Price gt 3.5
or	Logical or	/Products?$filter=Price le 3.5 or Price gt 200
not	Logical negation	/Products?$filter=not endswith(Description,'milk')
Arithmetic Operators		
add	Addition	/Products?$filter=Price add 5 gt 10
sub	Subtraction	/Products?$filter=Price sub 5 gt 10
mul	Multiplication	/Products?$filter=Price mul 2 gt 2000
div	Division	/Products?$filter=Price div 2 gt 4
mod	Modulo	/Products?$filter=Price mod 2 eq 0

Функции OData для фильтрации

Function	Example
String Functions	
bool substringof(string searchString, string searchInString)	substringof('Alfreds',CompanyName)
bool endswith(string string, string suffixString)	endswith(CompanyName,'Futterkiste')
bool startswith(string string, string prefixString)	startswith(CompanyName,'Alfr')
int length(string string)	length(CompanyName) eq 19
int indexof(string searchInString, string searchString)	indexof(CompanyName,'lfreds') eq 1
string replace(string searchInString, string searchString, string replaceString)	replace(CompanyName,' ', '') eq 'AlfredsFutterkiste'
string substring(string string, int pos)	substring(CompanyName,1) eq 'lfreds Futterkiste'
string substring(string string, int pos, int length)	substring(CompanyName,1, 2) eq 'lf'
string tolower(string string)	tolower(CompanyName) eq 'alfreds futterkiste'
string toupper(string string)	toupper(CompanyName) eq 'ALFREDS FUTTERKISTE'
string trim(string string)	trim(CompanyName) eq 'Alfreds Futterkiste'
string concat(string string1, string string2)	concat(concat(City,', '), Country) eq 'Berlin, Germany'
Date Functions	
int day(DateTime datetimeValue)	day(BirthDate) eq 8
int hour(DateTime datetimeValue)	hour(BirthDate) eq 1
int minute(DateTime datetimeValue)	minute(BirthDate) eq 0
int month(DateTime datetimeValue)	month(BirthDate) eq 12
int second(DateTime datetimeValue)	second(BirthDate) eq 0
int year(DateTime datetimeValue)	year(BirthDate) eq 1948

Math Functions	
double round(double doubleValue)	round(Freight) eq 32
decimal round(decimal decimalValue)	round(Freight) eq 32
double floor(double doubleValue)	floor(Freight) eq 32
decimal floor(decimal datetimeValue)	floor(Freight) eq 32
double ceiling(double doubleValue)	ceiling(Freight) eq 33
decimal ceiling(decimal datetimeValue)	ceiling(Freight) eq 33
Type Functions	
bool IsOf(type value)	isof('NorthwindModel.Order')
bool IsOf(expression value, type targetType)	isof(ShipCountry,'Edm.String')

Получение сессии клиента API

Для того чтобы можно было отправлять запросы напрямую со страницы, а не через web-сервер клиента, необходимо получить сессионный ключ, с которым можно будет обращаться в API. Для получения ключа необходимо выполнить GET-запрос

```
https://api.smart-bilet.ru/adm/api/Auth/GetSessionKey
```

Ответ сервера:

```
{"session":"37cf5d4c066a48cd91786b0042dc48af"}
```

Полученный сессионный ключ необходимо будет передавать в каждом запросе в cookie с именем `sessionKey` либо в параметре URI с именем `sessionKey`.

```
https://api.smart-bilet.ru/adm/api/Events?sessionKey=37cf5d4c066a48cd91786b0042dc48af
```

После получения сессионного ключа, предыдущий ключ будет недействительным.

Сессионный ключ остается неизменным достаточно продолжительное время. Однако, он может быть изменен в любой момент без предупреждения. В случае получения кодов 401, 402 или 403, необходимо повторно получить ключ.

Города

Для получения списка городов, необходимо выполнить GET-запрос в метод Towns.

Пример запроса:

```
https://api.smart-bilet.ru/adm/api/Towns
```

Пример ответа:

```
{
   "odata.metadata":"https://api.smart-bilet.ru/adm/api/$metadata#Towns",
   "value":[
    {
       "Id":1,
       "Title":"Санкт-Петербург",
       "KladrId":"7800000000000"
    },
```

```
    {
       "Id":2,
       "Title":"Хельсинки, Helsinki",
       "KladrId":null
   ]
}
```

Информацию об активных событиях в городе можно получить добавив к параметру `$expand` поле `EventsInfo`.

Билетные кассы

Для получения списка касс, необходимо выполнить GET-запрос в метод TicketOffices.

Пример запроса:

```
https://api.smart-bilet.ru/adm/api/TicketOffices
```

Пример ответа:

```
{
 "odata.metadata":"https://api.smart-bilet.ru/adm/api/$metadata#TicketOffices",
 "value":[
  {
   "Title":"Центральная касса",
   "Type":1,
   "Address":"Санкт-Петербург, Сердобольская ул., д. 64, корп. 1А",
   "TownId":1,
   "Latitude":null,
   "Longitude":null
  }
 ]
}
```

Категории мероприятий

Список категорий

Для получения списка категорий мероприятий, необходимо выполнить GET-запрос в метод Categories.

Пример запроса:

```
https://api.smart-bilet.ru/adm/api/Categories?$select=Id,Title
```

Пример ответа:

```
{
 "odata.metadata":"https://api.smart-
 bilet.ru/adm/api/$metadata#Categories&$select=Id,Title",
 "value":[
  {
   "Id":1,
   "Title":"Фестивали"
  },
  {
   "Id":6,
   "Title":"Театр"
  },
```

```
    {
      "Id":8,
      "Title":"Активный отдых"
    },
    {
      "Id":13,
      "Title":"Шоу"
    },
    {
      "Id":28,
      "Title":"Кино"
    },
    {
      "Id":29,
      "Title":"Концерты"
    },
    {
      "Id":32,
      "Title":"Спорт"
    },
    {
      "Id":46,
      "Title":"Разное"
    },
    {
      "Id":51,
      "Title":"Детям"
    }
  ]
}
```


В ответе будет содержаться список корневых категорий, подкатегории можно получить добавив к параметру `$expand` поле `Childs`.

Одна категория

Пример запроса:

```
https://api.smart-bilet.ru/adm/api/Categories(6)?$select=Id,Title,Childs/Id,Childs/Title&
$expand=Childs
```

Пример ответа:


```
{
 "odata.metadata":"https://api.smart-
 bilet.ru/adm/api/$metadata#Categories/@Element&$select=Id,Title,Childs/Id,Childs/Title",
 "Childs":[
    {
      "Id":7,
      "Title":"Театр"
    },
    {
      "Id":14,
      "Title":"Спектакль*"
    },
    {
      "Id":15,
      "Title":"Творческий вечер*"
    },
    {
      "Id":19,
```

```
      "Title":"Балет*"
    },
    {
      "Id":20,
      "Title":"Опера - Оперетта*"
    },
    {
      "Id":21,
      "Title":"Концерт"
    },
    {
       "Id":23,
       "Title":"Музыкальный спектакль"
    },
    {
      "Id":24,
      "Title":"Мюзикл"
    },
    {
      "Id":25,
      "Title":"Балет на льду"
    },
    {
      "Id":26,
      "Title":"Лекция-концерт"
    },
    {
      "Id":38,
      "Title":"Драма - Комедия"
    },
    {
      "Id":60,
      "Title":"моноспектакль"
    }
  ],
  "Id":6,
  "Title":"Театр"
}
```


Мероприятия

Список актуальных мероприятий

Для получения списка мероприятий, необходимо выполнить GET-запрос в метод Actions.

Пример запроса:

```
https://api.smart-bilet.ru/adm/api/Actions?$expand=Description
```

Пример ответа:


```
{
  "odata.metadata":"http://localhost:65265/api/$metadata#Actions","value":[
    {
   "Description":{
      "Value":"<p>\"МЫ ЖИВЁМ, ПОД СОБОЮ НЕ ЧУЯ СТРАНЫ...\"</p><p>Триптих</p><p>Спектакль
      первый.</p><p>\"МАНДЕЛЬШТАМА НЕТ\"</p><p>По воспоминаниям Надежды
      Мандельштам</p><p>Сценическая композиция и постановка Олега ДМИТРИЕВА</p><p>Играет
      Галина Филимонова (актриса МДТ - Театра Европы)</p><p>В спектакле \"Мандельштама нет\"
      Авторский театр на основе документальной истории травли и уничтожения великого поэта
```

```
      Осипа Мандельштама исследует тему государственного террора, его механизмов и средств,
      главные из которых: подмена нравственных ценностей идеологическими догмами, лишение
      человека свободы мысли и совести, разрушение личности путём моральных и физических
      пыток, уничтожение духовной жизни, как таковой, с целью формирования человеческого
      общества, как абсолютно послушной государству \"народной массы\", пригодной к
      расходованию в любых целях и любых количествах по произволу государства.</p><p>В
      спектакле о времени тотального страха свидетельствует вдова поэта Надежда Мандельштам в
      попытке призвать себя и всех к покаянию за вольное и невольное соучастие в
      преступлениях государства против человека и человечности.</p><p>Авторы спектакля,
      который является первой частью триптиха \"Мы живём, под собою не чуя страны...\",
      обнаруживают черты \"века-волкодава\" в сегодняшнем дне, его тяжкое наследство  в себе
      самих и вслед за Надеждой Мандельштам утверждают: пытаясь идти в будущее, не осознав
      своего прошлого, мы обречены на возврат к тому обществу, в котором МАНДЕЛЬШТАМА
      НЕТ.</p><p>Премьера спектакля состоялась 28 декабря 2008 года</p><p>Постановка
      спектакля осуществлена при поддержке Комитета по культуре Санкт-Петербурга и Малого
      драматического театра - Театра Европы</p>"
   },
   "Id":2280,
   "Title":"Мандельштама нет",
   "Announcement":null,
   "AgeGroup":16,
   "CategoryId":6,
   "SubCategoryId":14,
   "PosterId":"YO7NkaT7bEBuUSnnR-nshQ",
   "PosterIds":[
     "YO7NkaT7bEBuUSnnR-nshQ"
   ],
   "IsPushkinsCard":false,
   "Integrations":[],
   "Restrictions":[]
   }
  ]
}
```


Категории и подкатегории можно получить добавив к параметру `$expand` поля `Category, SubCategory`.

Ближайшее актуальное событие можно получить добавив к параметру `$expand` поле `NearestEvent`.

Одно мероприятие

Пример запроса:

```
https://api.smart-bilet.ru/adm/api/Actions(2280)?$expand=Description
```

Пример ответа:


```
{
"odata.metadata":"https://api.smart-bilet.ru/adm/api/$metadata#Actions/@Element",
   "Description":{
      "Value":"<p>\"МЫ ЖИВЁМ, ПОД СОБОЮ НЕ ЧУЯ СТРАНЫ...\"</p><p>Триптих</p><p>Спектакль
      первый.</p><p>\"МАНДЕЛЬШТАМА НЕТ\"</p><p>По воспоминаниям Надежды
      Мандельштам</p><p>Сценическая композиция и постановка Олега ДМИТРИЕВА</p><p>Играет
      Галина Филимонова (актриса МДТ - Театра Европы)</p><p>В спектакле \"Мандельштама нет\"
      Авторский театр на основе документальной истории травли и уничтожения великого поэта
      Осипа Мандельштама исследует тему государственного террора, его механизмов и средств,
      главные из которых: подмена нравственных ценностей идеологическими догмами, лишение
      человека свободы мысли и совести, разрушение личности путём моральных и физических
      пыток, уничтожение духовной жизни, как таковой, с целью формирования человеческого
      общества, как абсолютно послушной государству \"народной массы\", пригодной к
      расходованию в любых целях и любых количествах по произволу государства.</p><p>В
      спектакле о времени тотального страха свидетельствует вдова поэта Надежда Мандельштам в
```

```
      попытке призвать себя и всех к покаянию за вольное и невольное соучастие в
      преступлениях государства против человека и человечности.</p><p>Авторы спектакля,
      который является первой частью триптиха \"Мы живём, под собою не чуя страны...\",
      обнаруживают черты \"века-волкодава\" в сегодняшнем дне, его тяжкое наследство  в себе
      самих и вслед за Надеждой Мандельштам утверждают: пытаясь идти в будущее, не осознав
      своего прошлого, мы обречены на возврат к тому обществу, в котором МАНДЕЛЬШТАМА
      НЕТ.</p><p>Премьера спектакля состоялась 28 декабря 2008 года</p><p>Постановка
      спектакля осуществлена при поддержке Комитета по культуре Санкт-Петербурга и Малого
      драматического театра - Театра Европы</p>"
   },
   "Id":2280,
   "Title":"Мандельштама нет",
   "Announcement":null,
   "AgeGroup":16,
   "CategoryId":6,
   "SubCategoryId":14,
   "PosterId":"YO7NkaT7bEBuUSnnR-nshQ",
   "PosterIds":[
     "YO7NkaT7bEBuUSnnR-nshQ"
   ],
   "IsPushkinsCard":true,
   "Integrations":[
      {
        "Name":"ProCulture",
        "ExternalId":"123"
      }
   ],
   "Restrictions":[]
}
```


Площадки

Список площадок

Для получения списка площадок, необходимо выполнить GET-запрос в метод Venues.

Пример запроса:

```
https://api.smart-bilet.ru/adm/api/Venues
```

Пример ответа:


```
{
   "odata.metadata":"https://api.smart-bilet.ru/adm/api/$metadata#Venues",
   "value":[
    {
      "Id":12,
      "Title":"Клуб \"A2\"",
      "Address":"Санкт-Петербург",
      "TownId":1,
      "PosterId":null,
      "Latitude":null,
      "Longitude":null
    },
    {
      "Id":13,
      "Title":"ЦПКиО им. С.М. Кирова",
      "Address":"Санкт-Петербург",
      "TownId":1,
      "PosterId":null,
```

```
      "Latitude":null,
      "Longitude":null
    }
   ]
}
```


Одна площадка

Пример запроса:

```
https://api.smart-bilet.ru/adm/api/Venues(12)?$expand=Description
```

Пример ответа:


```
{
 "odata.metadata":"https://api.smart-bilet.ru/adm/api/$metadata#Venues/@Element",
 "Description":{
    "Value":"<p style=\"text-align: justify;\">Возрожденный клуб «A2»: новое здание, новые
    площадки, новая музыка!</p>\r\n<p style=\"text-align: justify;\">В сентябре 2012 года
    произошло громкое событие в клубной жизни Северной столицы. После двухлетнего бездействия
    клуб <strong>A2</strong> снова открылся! На этот раз в здании бывшего Завода
    Полиграфических Машин расположились сразу две концертные площадки «A2»: большой «Мир»,
    рассчитанный на 5 000 человек, и «Спутник», куда смогут вместиться 1 500 зрителей. И это
    не считая баров, ресторанов и новых, еще только готовящихся к открытию
    площадок.</p>\r\n<p style=\"text-align: justify;\">Организаторы обещают нам лучший звук в
    городе, недорогие цены и музыкальный репертуар для тех, у кого действительно есть
    музыкальный вкус!</p>"
 },
 "Id":12,
 "Title":"Клуб \"A2\"",
 "Address":"Санкт-Петербург",
 "TownId":1,
 "PosterId":null,
 "Latitude":null,
 "Longitude":null
}
```


Залы

Список залов

Для получения списка залов, необходимо выполнить GET-запрос в метод Venues, с указанием идентификатора площадки, по которой требуется получить список залов.

```
https://api.smart-bilet.ru/adm/api/Venues(12)/Halls
```

Пример ответа:


```
{
   "odata.metadata":"https://api.smart-bilet.ru/adm/api/$metadata#VenueHalls",
   "value":[
    {
     "Id":2,
```

```
      "Title":"Главный зал"
    },
    {
      "Id":4,
      "Title":" Главный зал. Только входные
    }
   ]
}
```


Сектора

Список секторов

Для получения списка секторов, необходимо выполнить GET-запрос в метод VenueHalls, с указанием идентификатора зала, по которому необходимо получить список секторов.

```
https://api.smart-bilet.ru/adm/api/VenueHalls(2)/Sectors
```

Пример ответа:


```
{
   "odata.metadata":"https://api.smart-bilet.ru/adm/api/$metadata#Sectors",
   "value":[
    {
      "Id":1,
      "Type":1,
      "SvgFileId":"",
      "Name":"Сектор A"
    },{
      "Id":2,
      "Type":1,
      "SvgFileId":"",
      "Name":"Сектор B"
    },{
      "Id":3,
      "Type":1,
      "SvgFileId":"",
      "Name":"Сектор C"
    },{
      "Id":4,
      "Type":1,
      "SvgFileId":"",
      "Name":"Сектор D"
    },
    {
      "Id":5,
      "Type":1,
      "SvgFileId":"",
      "Name":"Сектор E"
    }
   ]
}
```


Идентификаторы секторов уникальны в пределах зала.

Места

Список мест

Для получения списка мест, необходимо выполнить GET-запрос в метод VenueHalls, с указанием идентификатора зала, по которому необходимо получить список мест, а также идентификатор сектора.

```
https://api.smart-bilet.ru/adm/api/VenueHalls(2)/Places?sectorId=1
```

Пример ответа:


```
{
  "odata.metadata":" https://api.smart-bilet.ru/adm/api/$metadata#Places",
  "value": [
    {
      "Id": 1,
      "SectorId": 1,
      "Loge": null,
      "Row": "1",
      "Seat": "1",
      "X": 225.259,
      "Y": 158.731
    },
    {
      "Id": 2,
      "SectorId": 1,
      "Loge": null,
      "Row": "1",
      "Seat": "2",
      "X": 225.259,
      "Y": 185.964
    },
    {
      "Id": 3,
      "SectorId": 1,
      "Loge": null,
      "Row": "1",
      "Seat": "3",
      "X": 225.259,
      "Y": 213.196
    },
    {
      "Id": 4,
      "SectorId": 1,
      "Loge": null,
      "Row": "1",
      "Seat": "4",
      "X": 225.259,
      "Y": 240.429
    },
    {
      "Id": 5,
      "SectorId": 1,
      "Loge": null,
      "Row": "1",
      "Seat": "5",
      "X": 225.259,
      "Y": 267.661
```

```
    },
    {
      "Id": 6,
      "SectorId": 1,
      "Loge": null,
      "Row": "1",
      "Seat": "6",
      "X": 225.259,
      "Y": 294.894
    }
  ]
}
```

Идентификаторы мест уникальны в пределах зала.

Поставщики билетов

Список поставщиков билетов

Для получения списка поставщиков билетов, необходимо выполнить GET-запрос в метод Providers.

Пример запроса:

```
https://api.smart-bilet.ru/adm/api/Providers
```

Пример ответа:

```
{
   "odata.metadata":" https://api.smart-bilet.ru/adm/api/$metadata#Providers",
   "value":[
      {
         "Id":27,
         "Title":"Культпросвет",
         "INN":"7838448636",
         "OfficialName":"ООО Культпросвет",
         "OfficialAdress":""
      },
      {
         "Id":28,
         "Title":"Лайв! Промоушен",
         "INN":"7805414335",
         "OfficialName":"ООО ЛАЙВ! Промоушен",
         "OfficialAdress":""
      }
   ]
}
```

События

Список событий

Для получения списка событий, необходимо выполнить GET-запрос в метод Events. Список выдаваемых событий ограничен по длине – в ответе всегда не более 50 событий, даже если в параметре `$top` указать большее значение.

В событиях также отображается в процентах величина возврата цены билета события **на момент запроса** - `ReturnTicketPart`. То есть если до события осталось более 10 дней, поле может содержать значение 100, если

1 день – 0. У большинства событий поле всегда имеет значение 100, что значит, что поставщик билетов не включил возможность частичного возврата. Это поле напрямую влияет на цену билетов при отмене заказа или возврате билета.

Пример запроса:

```
https://api.smart-bilet.ru/adm/api/Events
```

Пример ответа:


```
{
   "odata.metadata":" https://api.smart-bilet.ru/adm/api/$metadata#Events",
   "value":[
      {
         "Id":11653,
         "Date":"2014-07-31T19:00:00",
         "Duration":80,
         "VenueId":362,
         "VenueHallId":354,
         "ActionId":2280,
         "ProviderId":27,
         "MainTariffId":40,
         "TicketCount":100,
         "TicketType":1,
         "MinPrice":"500.00",
         "MaxPrice":"500.00",
         "SellOpened":true,
         "SvgFileId":null,
         "AllowEtickets":false,
         "AllowTickets":true,
         "MinCartPrice":"0.00",
         "MaxCartPrice":"500.00",
         "MinCartTickets":"0",
         "MaxCartTickets ":"10",
         "ReturnTicketPart":"100",
         "Integrations":[
            {
               "Name":"ProCulture",
               "ExternalId":"123"
            }
         ],
         "Restrictions":[]
      },
      {
         "Id":11654,
         "Date":"2014-07-31T19:00:00",
         "Duration":120,
         "VenueId":269,
         "VenueHallId":261,
         "ActionId":243,
         "ProviderId":27,
         "MainTariffId":40,
         "TicketCount":66,
         "TicketType":1,
         "MinPrice":"1000.00",
         "MaxPrice":"1000.00",
         "SellOpened":true,
         "SvgFileId":"4-IIdvT6smFuYNhgENOKpQ",
         "AllowEtickets":false,
         "AllowTickets":true,
```

```
        "MinCartPrice":"0.00",
        "MaxCartPrice":"500.00",
        "MinCartTickets":"0",
        "MaxCartTickets ":"10",
        "ReturnTicketPart":"100",
        "Integrations":[
           {
             "Name":"ProCulture",
             "ExternalId":"123"
           }
        ],
        "Restrictions":[]
     }
   ]
}
```


Список событий возможно получить и из других мест. При этом, будет выведен список событий, отфильтрованный по соответствующему параметру.

Из города:

```
https://api.smart-bilet.ru/adm/api/Towns(1)/Events
```

Из площадки:

```
https://api.smart-bilet.ru/adm/api/Venues(269)/Events
```

Из зала площадки:

```
https://api.smart-bilet.ru/adm/api/VenueHalls(261)/Events
```

Из категории мероприятий:

```
https://api.smart-bilet.ru/adm/api/Categories(6)/Events
```

Из мероприятия:

```
https://api.smart-bilet.ru/adm/api/Actions(243)/Events
```

Одно событие

При получение события по его идентификатору, возможно указание в параметре `$expand` свойства `Sectors` (при выгрузке списка событий, указание этого параметра ни к чему не приведет).

Пример запроса:

```
https://api.smart-bilet.ru/adm/api/Events(11654)?$expand=Sectors
```

Пример ответа:


```
{
 "odata.metadata":"https://api.smart-bilet.ru/adm/api/$metadata#Events/@Element",
 "Sectors":[
```

```
    {
      "Id":1,
      "Type":2,
      "SvgFileId":null,
      "Name":"входной",
      "Count":100,
      "MinPrice":"500.00",
      "MaxPrice":"500.00"
    }
  ],
  "Id":11653,
  "Date":"2014-07-31T19:00:00",
  "Duration":80,
  "VenueId":362,
  "ActionId":2280,
  "ProviderId":27,
  "TicketCount":100,
 "TicketType":1,
  "MinPrice":"500.00",
  "MaxPrice":"500.00",
  "SellOpened":true,
  "SvgFileId":null,
  "AllowEtickets":false,
  "AllowTickets":true,
  "MinCartPrice":"0.00",
  "MaxCartPrice":"500.00",
  "MinCartTickets":"0",
  "MaxCartTickets ":"10",
  "ReturnTicketPart":"100",
  "Integrations":[],
  "Restrictions":[]
}
```


Также, возможно раскрытие описания мероприятия и площадки

```
https://api.smart-bilet.ru/adm/api/Events(11654)?$expand=Action/Description,
Venue/Description
```

Свободные билеты

Список свободных билетов с местами на событие можно получить следующим образом

```
https://api.smart-bilet.ru/adm/api/Events(11654)/FreeTickets
```

Пример ответа


```
{
  "odata.metadata":"https://api.smart-bilet.ru/adm/api/$metadata#Tickets",
  "value":[
    {
      "Tariffs":[
        {
          "Id":81,
          "Name":"БазовыйБК3",
          "Price":"1000.00",
          "ServicePrice":"0"
        }
      ],
      "Id":1463026,
```

```
      "Price":"1000.00",
      "ServicePrice":"0.00",
      "SectorId":1,
      "Sector":"Партер",
      "PlaceId":934,
      "Loge":null,
      "Row":"14",
      "Seat":"16"
    },
    {
      "Tariffs":[
        {
          "Id":81,
          "Name":"БазовыйБК3",
          "Price":"1000.00",
          "ServicePrice":"0"
        }
      ],
      "Id":1463027,
      "Price":"1000.00",
      "ServicePrice":"0.00",
      "SectorId":1,
      "Sector":"Партер",
      "PlaceId":935,
      "Loge":null,
      "Row":"14",
      "Seat":"17"
    }
  ]
}
```


Список свободных входных билетов можно получить следующим образом

```
https://api.smart-bilet.ru/adm/api/Events(11653)/FreeAdmissions
```

Пример ответа


```
{
  "odata.metadata":"https://api.smart-bilet.ru/adm/api/$metadata#AdmissionTickets",
  "value":[
    {
    "Tariffs":[
      {
        "Id":459,
        "Name":"Базовый",
        "Price":"500.00",
        "ServicePrice":"0"
      }
    ],
    "SectorId":1,
    "Name":"входной",
    "Count":100,
    "Price":"500.00"
    }
  ]
}
```


Промоакции

Список доступных промоакций можно получить следующим образом

```
https://api.smart-bilet.ru/adm/api/Events(11654)/PromoActions
```

Пример ответа


```
{
  "odata.metadata": "https://api.smart-
bilet.ru/adm/api/$metadata#Collection(Ekassir.TicketSystem.Web.API.Models.PromoAction)",
  "value": [
    {
      "PromoId": 1,
      "Description": "PromoAction By price from 20 to 1000",
      "EndTime": "2016-03-31T00:00:00",
      "EndTimeOfDay": null,
      "Name": "ByPrice",
      "Permanent": false,
      "Rules": [
        {
          "Discount": "15.00",
          "DiscountType": "Percent",
          "From": "20.00",
          "Round": false,
          "RuleType": "ByTicketsPrice",
          "To": "1000.00"
        }
      ],
      "StartTime": "2016-02-18T00:00:00",
      "StartTimeOfDay": null,
      "WeekDays": "None",
      "WholeDay": true
    }
  ]
}
```


Корзина пользователя

Создание сессии пользователя

Для создания сессии пользователя необходимо выполнить POST-запрос с пустым телом запроса:

```
https://api.smart-bilet.ru/adm/api/Users/CreateSession
```

Пример ответа


```
{
 "odata.metadata":"https://api.smart-bilet.ru/adm/api/$metadata#Users/@Element",
 "Session":"c88ed9b37caf4e259f993a95ed0e7f0f"
}
```


Далее, для примеров, при обращении к корзине будет использована эта сессия пользователя.

Поместить билет с местом в корзину

Для того чтобы поместить билет в корзину пользователя, необходимо выполнить POST-запрос с данными о Id события, Id билета и Id тарифа:

```
https://api.smart-bilet.ru/adm/api/Cart('c88ed9b37caf4e259f993a95ed0e7f0f')/PutTicket
```

```
POST-data:
{ EventId: 11654, TicketId: 1463026, TariffId: 81 }
```

Пример ответа


```
{
 "odata.metadata":"https://api.smart-bilet.ru/adm/api/$metadata#Tickets/@Element",
 "Id":1463026,
 "Price":"1000.00",
 "ServicePrice":"0.00",
 "SectorId":1,
 "Sector":"Партер",
 "PlaceId":934,
 "Loge":null,
 "Row":"14",
 "Seat":"16",
 "PromoActionId":null
}
```


Поместить входной билет в корзину

Для того чтобы поместить билет в корзину пользователя, необходимо выполнить POST-запрос с данными о Id события, Id сектора, Id тарифа, цене билета, кол-ве добавляемых входных билетах:

```
https://api.smart-bilet.ru/adm/api/Cart('c88ed9b37caf4e259f993a95ed0e7f0f')/PutAdmTicket
```

```
POST-data:
{ EventId: 11653, SectorId: 1, TariffId: 459, Price: "500.00", Count: 2 }
```

Пример ответа


```
{
   "odata.metadata":"https://api.smart-bilet.ru/adm/api/$metadata#Tickets",
   "value":[
      {
       "Id":1463107,
       "Price":"500.00",
       "ServicePrice":"0.00",
       "SectorId":1,
       "Sector":"входной",
       "PlaceId":null,
       "Loge":null,
       "Row":null,
       "Seat":null,
       "PromoActionId":null
      },
      {
       "Id":1463108,
       "Price":"500.00",
       "ServicePrice":"0.00",
       "SectorId":1,
       "Sector":"входной",
       "PlaceId":null,
       "Loge":null,
       "Row":null,
       "Seat":null,
```

```
      "PromoActionId":null
     }
   ]
}
```

Удалить билет из корзины

Для того чтобы удалить билет из корзины пользователя (с местом или входной), необходимо выполнить POST-запрос с данными о Id билета:

```
https://api.smart-bilet.ru/adm/api/Cart('c88ed9b37caf4e259f993a95ed0e7f0f')/DeleteTicket
```

```
POST-data:
{ TicketId: 1463026 }
```

Пример ответа

```
{
 "odata.metadata":"https://api.smart-bilet.ru/adm/api/$metadata#Tickets/@Element",
 "Id":1463026,
 "Price":"1000.00",
 "ServicePrice":"0.00",
 "SectorId":1,
 "Sector":"Партер",
 "PlaceId":934,
 "Loge":null,
 "Row":"14",
 "Seat":"16",
 "PromoActionId":null
}
```

Удалить входной билет из корзины

Для того чтобы удалить входные билеты из корзины пользователя, необходимо выполнить POST-запрос с данными о Id события, Id сектора, Id тарифа, цене билета, кол-ве удаляемых входных билетах:

```
https://api.smart-bilet.ru/adm/api/Cart('c88ed9b37caf4e259f993a95ed0e7f0f')/DeleteAdmTicket
```

```
POST-data:
{ EventId: 11653, SectorId: 1, TariffId: 459, Price: "500.00", Count: 2 }
```

Пример ответа

```
{
   "odata.metadata":"https://api.smart-bilet.ru/adm/api/$metadata#Tickets",
   "value":[
      {
       "Id":1463107,
       "Price":"500.00",
       "ServicePrice":"0.00",
       "SectorId":1,
       "Sector":"входной",
       "PlaceId":null,
       "Loge":null,
       "Row":null,
```

```
      "Seat":null,
      "PromoActionId":null
    },
    {
      "Id":1463108,
      "Price":"500.00",
      "ServicePrice":"0.00",
      "SectorId":1,
      "Sector":"входной",
      "PlaceId":null,
      "Loge":null,
      "Row":null,
      "Seat":null,
      "PromoActionId":null
    }
  ]
}
```


Применить промоакцию к корзине

Для того чтобы применить промоакцию к корзине пользователя, необходимо выполнить POST-запрос с данными о Id промоакции или промокоде:

```
https://api.smart-bilet.ru/adm/api/Cart('c88ed9b37caf4e259f993a95ed0e7f0f')/ApplyPromoAction
```

```
POST-data:
{ PromoAction: { PromoCode: "DiscountPlease" } }
или:
{ PromoAction: { PromoActionId: "11", PromoActionRuleId: "44" } }
```

Пример ответа


```
{
  "odata.metadata": " https://api.smart-bilet.ru/adm/api/$metadata#Cart/@Element",
  "Tickets": [
    {
      "Id": 3362161,
      "Price": "370.00",
      "ServicePrice": "0.00",
      "SectorId": 7,
      "Sector": "Ярус 4",
      "PlaceId": 884,
      "Loge": "3",
      "Row": "1",
      "Seat": "2",
      "Tariffs": [
        {
          "Id": 668,
          "Name": "Базовый",
          "Sort": 0,
          "Price": "400.00",
          "ServicePrice": "0.00"
        }
      ],
      "OrderedTariffId": 668,
      "Barcode": "",
      "EventId": 33398,
      "PromoActionId":"11"
    }
  ],
```

```
  "UserSession": "c88ed9b37caf4e259f993a95ed0e7f0f",
  "Ttl": 844,
  "Count": 1,
  "Sum": "370.00",
  "OriginalSum": "400.00",
  "AppliedPromoAction": true,
  "PromoActionId": "11"
}
```

Удалить промоакцию из корзины

Для того чтобы удалить промоакцию, необходимо выполнить POST- запрос с пустым телом запроса:

```
https://api.smart-bilet.ru/adm/api/Cart('c88ed9b37caf4e259f993a95ed0e7f0f')/DeletePromoAction
```

Пример ответа

```
{
  "odata.metadata": " https://api.smart-bilet.ru/adm/api/$metadata#Cart/@Element",
  "Tickets": [
    {
      "Id": 3362161,
      "Price": "370.00",
      "ServicePrice": "0.00",
      "SectorId": 7,
      "Sector": "Ярус 4",
      "PlaceId": 884,
      "Loge": "3",
      "Row": "1",
      "Seat": "2",
      "Tariffs": [
        {
          "Id": 668,
          "Name": "Базовый",
          "Sort": 0,
          "Price": "400.00",
          "ServicePrice": "0.00"
        }
      ],
      "OrderedTariffId": 668,
      "Barcode": "",
      "EventId": 33398,
      "PromoActionId":null
    }
  ],
  "UserSession": "c88ed9b37caf4e259f993a95ed0e7f0f",
  "Ttl": 844,
  "Count": 1,
  "Sum": "400.00",
  "OriginalSum": null,
  "AppliedPromoAction": false,
  "PromoActionId": null
}
```

Просмотр корзины

Для просмотра корзины пользователя необходимо выполнить GET-запрос:

```
https://api.smart-bilet.ru/adm/api/Cart('c88ed9b37caf4e259f993a95ed0e7f0f')?$expand=Tickets
```

Для просмотра корзины пользователя с расширенной информацией о промоакциях необходимо выполнить GET-запрос:

```
https://api.smart-bilet.ru/adm/api/
Cart('c88ed9b37caf4e259f993a95ed0e7f0f')?$expand=Tickets/PromoAction
```

Пример ответа


```
{
 "odata.metadata":"https://api.smart-bilet.ru/adm/api/$metadata#Cart/@Element",
 "Tickets":[
    {
     "Id":1463026,
     "Price":"1000.00",
     "ServicePrice":"0.00",
     "SectorId":1,
     "Sector":"Партер",
     "PlaceId":934,
     "Loge":null,
     "Row":"14",
     "Seat":"16",
     "Tariffs": [
        {
          "Id": 668,
          "Name": "Базовый",
          "Sort": 0,
          "Price": "400.00",
          "ServicePrice": "0.00"
        }
      ],
     "OrderedTariffId":81,
     "EventId":11654,
     "PromoActionId": null
    },
    {
     "Id":1463111,
     "Price":"500.00",
     "ServicePrice":"0.00",
     "SectorId":1,
     "Sector":"входной",
     "PlaceId":null,
     "Loge":null,
     "Row":null,
     "Seat":null,
     "Tariffs": [
        {
          "Id": 668,
          "Name": "Базовый",
          "Sort": 0,
          "Price": "400.00",
          "ServicePrice": "0.00"
        }
      ],
     "OrderedTariffId":459,
     "EventId":11653   ,
     "PromoActionId": null
    },
    {
     "Id":1463112,
     "Price":"500.00",
```

```
      "ServicePrice":"0.00",
      "SectorId":1,
      "Sector":"входной",
      "PlaceId":null,
      "Loge":null,
      "Row":null,
      "Seat":null,
      "Tariffs": [
         {
           "Id": 668,
           "Name": "Базовый",
           "Sort": 0,
           "Price": "400.00",
           "ServicePrice": "0.00"
         }
       ],
      "OrderedTariffId":459,
      "EventId":11653 ,
      "PromoActionId": null
    }
  ],
  "UserSession":"c88ed9b37caf4e259f993a95ed0e7f0f",
  "Ttl":250,
  "Count":3,
  "Sum":"2000.00",
  "OriginalSum":null,
  "AppliedPromoAction": false,
  "PromoActionId": null
}
```


Доступные способы доставки

При наличии билетов в корзине, можно получить доступные способы доставки билетов. Способ доставки используется при создании брони. Для получения доступных способов доставки, необходимо отправить POST запрос с указанием `sessionId` в метод api: `Cart('sessionId')/GetDeliveryTypes`.

```
https://api.smart-bilet.ru/adm/api/Cart('c88ed9b37caf4e259f993a95ed0e7f0f')/GetDeliveryTypes
```

Пример ответа


```
{
  "odata.metadata":"https://api.smart-bilet.ru/adm/api$metadata#EnumValues",
  "value":[
    {
      "Id":1,
      "Title":"Получение в кассе",
      "Description":"Билеты можно будет получить в кассе или терминале"
    },
    {
      "Id":2,
      "Title":"Электронные билеты",
      "Description":"Билеты будут доступны сразу после оплаты. Доставка или печать на бланках
      не требуется."
    },
    {
      "Id":3,
      "Title":"Доставка курьером",
      "Description":"Билеты будут доставлены курьером."
    }
   ]
```

```
}
```

Доступные способы оплаты

При наличии билетов в корзине, а также при известном способе доставки, можно получить доступные способы оплаты билетов. Способ оплаты (так же, как и способ доставки) используется при создании брони. Для получения доступных способов оплаты, необходимо отправить POST запрос с указанием `sessionId` и `deliveryType` в метод api: `Cart('sessionId')/GetPaymentTypes`.

```
https://api.smart-bilet.ru/adm/api/Cart('c88ed9b37caf4e259f993a95ed0e7f0f')/
GetPaymentTypes?deliveryType=1
```

Пример ответа

```
{
 "odata.metadata":"https://api.smart-bilet.ru/adm/api$metadata#EnumValues",
 "value":[
    {
     "Id":1,
     "Title":"Наличными",
     "Description":"Наличными в кассе, либо курьеру при выборе способа доставки - \"Доставка
     курьером\""
    },
    {
     "Id":2,
     "Title":"Банковской картой",
     "Description":"Банковской картой на сайте продаж через выбранный эквайринг"
    },
    {
     "Id":3,
     "Title":"Электронными деньгами",
     "Description":"Электронными деньгами на сайте продаж"
    }
  ]
}
```

Брони пользователя

Создание брони из билетов в корзине

Для создания брони из билетов в корзине, необходимо передать в API данные о пользователе, и о доставке (если таковая имело место быть). Отправляемый в API объект состоит из 2-х полей:

`{ Order: { … }, Delivery: { … } }`

Свойство `Order` – это данные о заказе и заказчике, свойство `Delivery` – это данные о доставке. Если доставки нет, то передавать свойство `Delivery` не нужно, и запрос будет содержать только объект `{ Order: { … } }`.

Объект `Order` имеет следующие поля:

Параметр	Тип данных	Размер поля	Описание
ClientName*	string	200	Наименование клиента
Email*	string	200	E-mail клиента

PhoneNumber*	string	200	Телефон клиента
Comment	string	300	Комментарий к броне
PaymentType*	tinyint (*enum*)		Тип платежа (1 – Наличными, 2 – Банковской картой, 3 – Электронными деньгами)
Delivery	Delivery		Данные о доставке
DeliveryType*	tinyint (*enum*)		Тип доставки (1 – Самовыкуп на кассе, 2 – Электронный билет, 3 – Доставка курьером)
UtmTags	UtmTagsInfo		Данные о рекламной компании

** - отмечены поля, обязательные к заполнению*

Объект `Delivery` имеет следующие поля:

Параметр	Тип данных	Размер поля	Описание
DeliverFrom*	DateTime		Желаемая дата доставки (строка в формате YYYY-MM-dd HH:mm:ss)
DeliverTo*	DateTime		Желаемая дата доставки (строка в формате YYYY-MM-dd HH:mm:ss)
TownId*	int		Id города доставки
Street*	string	200	Улица, проспект, переулок и т.п. Вводится не только наименование, но и само сокращение «ул.», «пр-кт» или что-то подобное. Например, «ул. Ленина»
House*	string	50	Номер дома, включая литеру и/или корпус, если таковые имеются
Flat	string	50	Номер квартиры

** - отмечены поля, обязательные к заполнению*

Объект `UtmTagsInfo` имеет следующие поля:

Параметр	Тип данных	Размер поля	Описание
UtmSource	string	200	Источник трафика (Google, Яндекс, Email, Blog)
UtmMedium	string	200	Тип трафика (PPC, КМС, ретаргетинг, рассылка, пост, баннер)
UtmCampaign	string	200	Название рекламной кампании или рекламного объявления
UtmContent	string	200	Содержание, помогает различать объявления, содержащие одинаковые ключевые слова, может быть полезно для теста объявлений
UtmTerm	string	200	Ключевое слово
UtmReferrer	string	200	Источник перехода

Запрос на создание брони:

```
https://api.smart-bilet.ru/adm/api/Cart('c88ed9b37caf4e259f993a95ed0e7f0f')/CreateOrder
```

```
POST-data:
{ Order : { ClientName : "Василий Пупкин", Email : "vasya@mail.ru", PhoneNumber :
"+71234567890", PaymentType : 1, DeliveryType : 3 }, Delivery : { TownId : 1, Street : "ул.
Пети Пупкина", House : "3", Flat : "12", DeliverFrom : "2014-07-07 16:00:00", DeliverTo :
"2014-07-07 23:00:00"} }
```

Пример ответа

```
{
   "odata.metadata":"https://api.smart-bilet.ru/adm/api/$metadata#Orders",
   "value":[
    {
     "Id":99880,
     "InputTimeUTC":"2014-07-07T11:32:43.62Z",
     "CancelTimeUTC":"2014-07-07T11:47:43.62Z",
     "ClientName":"Test",
     "Email":"sdsd@sdsd.ru",
     "PhoneNumber":"11111",
     "PaymentType":1,
     "DeliveryType":3,
     "TicketCount":4,
     "TicketsPrice":"2500.00",
     "ServicePrice":"0.00",
     "DeliveryPrice":"0.00",
     "FullPrice":"2500.00",
     "Comment":null,
     "State":1
    }
   ]
}
```


Важно понимать, что при передаче не доступного способа доставки или оплаты, создание брони будет невозможно. Например, не поддерживается передача PaymentType = 1 (наличные) с DeliveryType = 2 (электронные билеты).

Список броней

Для получения списка броней пользователя, необходимо отправить запрос в контроллер `Orders`, передав при этом `userSession`. Если необходимо получить только одну бронь, то в запросе необходимо передавать параметр `orderId` с идентификатором необходимой брони.

```
https://api.smart-bilet.ru/adm/api/Orders('c88ed9b37caf4e259f993a95ed0e7f0f')?
$expand=Tickets
```

Пример ответа


```
{
   "odata.metadata":"https://api.smart-bilet.ru/adm/api/$metadata#Orders",
   "value":[
    {
     "Tickets":[
        {
           "Id":1463026,
           "Price":"1000.00",
           "ReturnPrice":"0.00",
           "ServicePrice":"0.00",
           "SectorId":1,
           "Sector":"Партер",
           "PlaceId":934,
           "Loge":null,
           "Row":"14",
           "Seat":"16",
           "OrderedTariffId":81,
           "EventId":11654,
           "PromoActionId": null
        },
        {
```

```
          "Id":1463112,
          "Price":"500.00",
          "ReturnPrice":"0.00",
          "ServicePrice":"0.00",
          "SectorId":1,
          "Sector":"входной",
          "PlaceId":null,
          "Loge":null,
          "Row":null,
          "Seat":null,
          "OrderedTariffId":459,
          "EventId":11653,
          "PromoActionId": null
       },
       {
          "Id":1463113,
          "Price":"500.00",
          "ReturnPrice":"0.00",
          "ServicePrice":"0.00",
          "SectorId":1,
          "Sector":"входной",
          "PlaceId":null,
          "Loge":null,
          "Row":null,
          "Seat":null,
          "OrderedTariffId":459,
          "EventId":11653,
          "PromoActionId": null
       },
       {
          "Id":1463114,
          "Price":"500.00",
          "ReturnPrice":"0.00",
          "ServicePrice":"0.00",
          "SectorId":1,
          "Sector":"входной",
          "PlaceId":null,
          "Loge":null,
          "Row":null,
          "Seat":null,
          "OrderedTariffId":459,
          "EventId":11653,
          "PromoActionId": null
       }
    ],
    "Delivery":{
     "DeliverFromUTC":"2014-07-07T12:00:00Z",
     "DeliverToUTC":"2014-07-07T19:00:00Z",
     "TownId":1,
     "Street":"ул. Пети Пупкина",
     "House":"222",
     "Flat":"12"
    },
    "Id":99880,
    "InputTimeUTC":"2014-07-07T11:32:43.62Z",
    "CancelTimeUTC":"2014-07-07T11:47:43.62Z",
    "ClientName":"Василий Пупкин",
    "Email":"vasya@mail.ru",
    "PhoneNumber":"+71234567890",
    "PaymentType":1,
    "DeliveryType":3,
    "TicketCount":4,
    "TicketsPrice":"2500.00",
```

```
      "ServicePrice":"0.00",
      "DeliveryPrice":"0.00",
      "FullPrice":"2500.00",
      "Comment":null,
      "State":1
    },
    {
      "Tickets":[
      ],
      "Id":99883,
      "InputTimeUTC":"2014-07-08T05:11:48.403Z",
      "CancelTimeUTC":"2014-07-08T05:26:48.4Z",
      "ClientName":"Василий Пупкин",
      "Email":"vasya@mail.ru",
      "PhoneNumber":"+71234567890",
      "PaymentType":1,
      "DeliveryType":1,
      "TicketCount":2,
      "TicketsPrice":"1000.00",
      "ServicePrice":"0.00",
      "DeliveryPrice":"0.00",
      "FullPrice":"1000.00",
      "Comment":null,
      "State":5
    }
  ]
}
```


Если бронь оплачена (State = 2), то в ответе будет содержаться информация о штрихкоде в каждом электронном билете (поле Barcode объекта CartTicket)

Оплата брони

На сайте продаж клиент теоретически может оплатить электронные или обычные билеты. Саму логику оплаты сайт должен реализовывать самостоятельно. В api следует посылать запрос на оплату брони только после того, как клиент оплатит бронь через эквайринг.

Для оплаты брони, необходимо отправить запрос методом POST в метод api Orders('sessionId')/Pay, передав при этом userSession в качестве параметр запроса, а OrderId необходимо передавать в теле запроса.

Если необходимо, можно передать данные платежа/платежей в массиве Payments (необязательный параметр).

```
https://api.smart-bilet.ru/adm/api/Orders('c88ed9b37caf4e259f993a95ed0e7f0f')/Pay
POST-data:
{
  OrderId: 99880,
  Payments: [{
    IsPushkinsCard:true,
    Amount:"2500.00",
    PaymentNumber:"some payid",
    RRN:"some rrn",
    RspCode:1
    RspMessage:"success"
  }]
}
```

Пример ответа

```
{
   "odata.metadata":"https://api.smart-bilet.ru/adm/api/$metadata#Orders",
   "value":[
    {
     "Delivery":{
      "DeliverFromUTC":"2014-07-07T12:00:00Z",
      "DeliverToUTC":"2014-07-07T19:00:00Z",
      "TownId":1,
      "Street":"ул. Пети Пупкина",
      "House":"222",
      "Flat":"12"
     },
     "Id":99880,
     "InputTimeUTC":"2014-07-07T11:32:43.62Z",
     "CancelTimeUTC":"2014-07-07T11:47:43.62Z",
     "ClientName":"Василий Пупкин",
     "Email":"vasya@mail.ru",
     "PhoneNumber":"+71234567890",
     "PaymentType":1,
     "DeliveryType":3,
     "TicketCount":4,
     "TicketsPrice":"2500.00",
     "ServicePrice":"0.00",
     "DeliveryPrice":"0.00",
     "FullPrice":"2500.00",
     "Comment":null,
     "State":2
}
```


Отмена брони

Для отмены брони, необходимо отправить запрос методом POST в метод api `Orders('sessionId')/Cancel`, передав при этом `userSession` в качестве параметр запроса, а `OrderId` необходимо передавать в теле запроса. Можно также указать причину отмены, в параметре `Reason` (поле не обязательное).

Необходимо учитывать, что при отмене автоматически пересчитывается стоимость возвращенных билетов, и она может быть не 100%. Можно выяснить это, проверив поле Event/ReturnTicketPart перед отменой брони или запросив проверку брони после её отмены.

```
https://api.smart-bilet.ru/adm/api/Orders('c88ed9b37caf4e259f993a95ed0e7f0f')/Cancel
POST-data:
{ OrderId: 99880, Reason: 'Отказ клиента' }
```

При успешной отмене в ответ придет пустое сообщение с кодом состояния `200` (OK).

После этого рекомендуется выполнить проверку брони.

Возврат билетов оплаченной брони

Для возврата выборочно билетов брони, необходимо отправить запрос методом POST в метод api `Orders('sessionId')/ReturnTickets`, передав при этом `userSession` в качестве параметра запроса.

Тело запроса:

`OrderId`. При отсутствующем поле или значении 0 вернётся HTTP код NotFound. Если заказ не оплачен вернётся HTTP код BadRequest "Заказ не оплачен.".

`Tickets`. Массив билетов в заказе. Если не указана стоимость билета вернётся HTTP код BadRequest "Стоимость билета не указана.".

В отличие от метода отмены брони в методе возврата билетов стоимость возвращённых билетов указывается самостоятельно.

```
https://api.smart-bilet.ru/adm/api/Orders('c88ed9b37caf4e259f993a95ed0e7f0f')/ReturnTickets
POST-data:
{
      "OrderId": 99880,
      "Tickets":[
       {
        "Id":99883,
        "Price":"1000.00",
       }
      ]
}
```

При успешной отмене в ответ придет пустое сообщение с кодом состояния `200` (OK).

После этого рекомендуется выполнить проверку брони.

Добавление атрибута брони/билета

Атрибуты служат для добавления дополнительной текстовой информации о заказе – UTM-метки, внешний идентификатор, дополнительные контактные данные. В api необходимо посылать запрос на добавление атрибутов билетов только после того, как будет создана бронь. Если в объекте уже содержится атрибут с названием добавляемого атрибута, то значение данного атрибута перезапишется.

Для добавления атрибутов необходимо отправить запрос методом POST в метод api `Orders('sessionId')/AddAttributes`, передав при этом `userSession` в качестве параметр запроса, а `OrderId, Attributes` необходимо передавать в теле запроса. Для добавления атрибута билета необходимо дополнительно указать `TicketId`.

```
https://api.smart-bilet.ru/adm/api/Orders('c88ed9b37caf4e259f993a95ed0e7f0f')/AddAttributes
POST-data для атрибутов брони:
{ OrderId: 99880,
Attributes: [{Name:"attr1",Value:"attr1Val"}, {Name:"attr2",Value:"attr2Val"}] }
POST-data для атрибутов билета:
{ OrderId: 99880, TicketId:2424870,
Attributes: [{Name:"attr1",Value:"attr1Val"}, {Name:"attr2",Value:"attr2Val"}] }
```

При успешном обновлении атрибутов в ответ придёт список атрибутов объекта.

Пример ответа

```
{
  "odata.metadata": "https://localhost:44302/api/$metadata#Attributes",
  "value": [
    {
      "Id": "90712a56-df0e-4315-b593-8ec3462354ef", // игнорировать
      "Name": "attr1",
      "Value": "attr1Val"
    },
    {
      "Id": "61dba326-fcdd-4c4a-89b2-28509f1a7844",
```

```
      "Name": "attr2",
      "Value": " attr2Val"
    }
  ]
}
```

Отправка email о заказе покупателю

Система автоматически отправляет покупателю email-сообщения о создании, оплате и отмене заказа в случае, если настроен почтовый сервер и шаблоны сообщений (настройка производится отдельно в билетной системе).

Для повторной отправки необходимо отправить запрос методом POST в метод api `Orders('sessionId')/SendEmail`, передав при этом `userSession` в качестве параметр запроса, а `OrderId, MessageType` (Тип email-сообщения (1 – о создании брони, 2 – об оплате брони (с вложениями pdf), 3 – об отмене брони) необходимо передавать в теле запроса.

```
https://api.smart-bilet.ru/adm/api/Orders('c88ed9b37caf4e259f993a95ed0e7f0f')/SendEmail
POST-data для повторной отправки сообщения о создании брони:
{ OrderId: 99880, MessageType: 1 }
```

При успешном обновлении атрибутов в ответ придёт пустое сообщение с кодом состояния `200` (OK).

Получение файлов

Получение постера

У мероприятий и площадок есть поле `PosterId`, значение которого можно использовать для получение постера. Формат URI следующий: `/Media/{width:int}x{height:int}/{id}`, где `{width}, {height}` – ширина и высота постера (целое число), `{id}` – идентификатор постера.

```
https://api.smart-bilet.ru/adm/api/Media/200x200/YO7NkaT7bEBuUSnnR-nshQ
```

Вернет изображение размером 200 на 200 пикселей.

Получение svg-файлов залов и секторов

Для получения файлов используется идентификатор svg-файла у события и сектора – поле SvgFileId.

```
https://api.smart-bilet.ru/adm/api/Media/4-IIdvT6smFuYNhgENOKpQ
```

Получение штрихкода

После того, как были получены билеты из запроса «Список броней», если бронь оплачена, то у билета будет доступен штрихкод. Штрихкод можно использовать для прохода на мероприятие, если на нем используется система контроля доступа. Для получения самого изображения штрихкода используется метод `api/Media/Barcode/{barcode_value}`

```
https://api.smart-bilet.ru/adm/api/media/barcode/002055763496
```

Вернет изображение размером 300 на 100 пикселей с одномерным штрихкодом.

В данном запросе также можно указать следующие дополнительные параметры (передавать параметры необходимо в URI запроса)

Параметр	Тип данных	Значение по-умолчанию	Описание
width	int	300	Ширина
height	int	100	Высота
type	int	1	Тип штрихкода: 1 – одномерный штрихкод (Code 128) 2 – двумерный штрихкод (QR-код)
pure	bool	true	Управление выводом значения штрихкода при создании изображения штрихкода. true – не выводится на штрихкоде false – выводится на штрихкоде

Получение PDF-файла заказа

Если бронь оплачена и настроены шаблоны у всех билетов, у заказа будет доступна возможность загрузки единого PDF-файла, каждая страница которого соответствует отдельному билету. Штрихкоды билетов в файле можно использовать для прохода на мероприятие, если на нем используется система контроля доступа. Для получения файла используется метод `api/Media/Pdf/{order_id}`

```
https://api.smart-bilet.ru/adm/api/media/pdf/99880
```

Получение PDF-файла билета

Если бронь оплачена и настроен шаблон билета, у билета будет доступна возможность загрузки PDF-файла. Шаблон билета должен быть создан и настроен в билетной системе! Штрихкод в файле можно использовать для прохода на мероприятие, если на нем используется система контроля доступа. Для получения файла используется метод `api/Media/Pdf/{order_id}/{ticket_id}`

```
https://api.smart-bilet.ru/adm/api/media/pdf/99880/1463114
```

Ошибки API

Формат ответа

При возникновении исключительной ситуации, сервер ответит HTTP-кодом 400 – Bad Request и в теле запроса будет содержаться объект ODataError, в котором будет содержаться код ошибки и сообщение об ошибке.

```
{
 "odata.error":{
  "code":"Цифровой код ошибки",
  "message":{
   "lang":"en-US",
   "value":"Сообщение об ошибке"
```

```
    }
  }
}
```

Коды ошибок

Код	Type	Описание
1	BadRequest	Неверные параметры запроса
2	BadOperation	Неверная операция
3	ServerError	Ошибка сервера
4	GatewayError	Ошибка в шлюзе
5	PointIsBlocked	Точка заблокирована
6	ClientNotFound	Клиент не найден
7	OrderNotFound	Заказ не найден
8	TariffNotFound	Не найден тариф, по которому бронировать билет
9	TicketNotFound	Билет не найден
10	TicketNotEnough	Не хватает кол-во входных для бронирования
14	OrderStateInvalid	Статус брони не валидный для данного запроса
15	OrderTicketsCountTooLow	Слишком малое кол-во билетов в броне
16	OrderTicketsCountTooHigh	Слишком большое кол-во билетов в броне
17	OrderPriceTooLow	Слишком низкая стоимость брони
18	OrderPriceTooHigh	Слишком высокая стоимость брони
19	TicketStateInvalid	Статус билета не валидный для данного запроса
20	PointCantAccessToEvent	У точки продаж нет доступа к событию
21	PointCantSellTicketsOnEvent	Точка продаж не может продавать билеты на данное событие
22	EventSalesClosedByOwner	У события закрыты продажи поставщиком
23	EventSalesClosedByDealer	У события закрыты продажи дилером
24	DealerMainFundIsMissed	У дилера не настроен основной фонд
25	MissedTicketPatternAtEvent	Не настроен шаблон билета у события (при запросе PDF)
26	TicketAlreadyChecked	Билет уже отмечен (при проходе в зал)
28	ControllerTimeExceed	Время, до которого можно пройти на событие, уже наступило (при проходе в зал)
29	PointMustBeAController	Точка должна быть контролером (при проходе в зал)
32	TicketCanNotReturnInGateway	Невозможно вернуть билет в шлюзе
34	PaymentAmountIsInvalid	Сумма платежа неверная
37	InCartMissedAllowETicketEvents	В корзине отсутствуют события с возможностью продажи электронных билетов
38	DeliveryMustSpecified	Необходимо указать данные о доставке
40	PromoActionInCart	Есть примененная промоакция в билетах из корзины
40	NoServicePrice	Сервисный сбор равен нулю или не определен
42	CancelOrderOnSaleDay	Отмена заказа возможна только в день его создания
43	OnlyOrderPointCanCancelOrder	Отмена заказа возможна только на той точке, на которой был создан заказ
44	InCartMixedGatewayTickets	В корзине билеты из разных шлюзов или одновременно локальные билеты и билеты из шлюза
47	PromoActionNotApplied	Промоакция не применена
50	CantReturnUsedTicket	Невозможно вернуть использованный билет
51	AcceptedPaymentsNotFound	Не найдено принятых платежей в заказе
52	ConfirmedPaymentsNotFound	Не найдено подтвержденных платежей в заказе
53	PaymentsAlreadyConfirmed	Все платежи в заказе уже подтверждены
54	PaymentsAlreadyFinalized	Все платежи в заказе уже завершены
55	GatewayMultipleOrdersNotAllowed	Невозможно оплатить в шлюзе родительский заказ

Объекты API

Город (Town)

Параметр	Тип данных	Размер поля	Описание
Id	int		Id города
Title	string	200	Наименование города
KladrId	string	20	Код КЛАДР города
EventsInfo	EventsInfo		Информация о событиях в городе

Информация о событиях по умолчанию не будет в выборке. Его можно получить через `$expand=EventsInfo`
Например, так: `/api/Towns(1)?$expand=EventsInfo`

Билетная касса (TicketOffice)

Параметр	Тип данных	Размер поля	Описание
Title	string	200	Наименование билетной кассы
Type	tinyint (*enum*)		Тип кассы (1 – касса, 2 – терминал)
Address	string	500	Адрес кассы
TownId	int		Id города
Latitude	decimal	(12,9)	Широта расположения кассы на карте
Longitude	decimal	(12,9)	Долгота расположения кассы на карте
Town	Town		Город

Площадка (Venue)

Параметр	Тип данных	Размер поля	Описание
Id	int		Id площадки
Title	string	200	Наименование площадки
Description	Description		Описание
Address	string	500	Адрес площадки
TownId	int		Id города
PosterId	string	22	Id файла с постером
Latitude	decimal	(12,9)	Широта расположения площадки на карте
Longitude	decimal	(12,9)	Долгота расположения площадки на карте
Town	Town		Город

Описание площадки по умолчанию не будет в выборке. Его можно получить через `$expand=Description`, при этом само описание будет в поле `Description.Value`. Например, так:
`/api/Venues(15)?$expand=Description`

Зал (VenueHall)

Параметр	Тип данных	Размер поля	Описание
Id	int		Id зала
Title	string	200	Наименование зала

Сектор (Sector)

Параметр	Тип данных	Размер поля	Описание
Id	int		Id сектора
Type	tinyint (*enum*)		Тип сектора: 1 – сектор с местами, 2 – сектор без мест (входной)
Name	string	100	Наименование сектора
SvgFileId	string	22	Id файла сектора

Место (Place)

Параметр	Тип данных	Размер поля	Описание
Id	int		Id места
SectorId	int		Id сектора
Loge	string	50	Ложа
Row	string	50	Ряд
Seat	string	50	Место
X	float		X-координата места на схеме зала
Y	float		Y-координата места на схеме зала

Поставщик билетов (Provider)

Параметр	Тип данных	Размер поля	Описание
Id	int		Id поставщика билетов
Title	string	250	Наименование
INN	string	20	ИНН
OfficialName	string	200	Юридическое наименование
OfficialAdress	string	200	Юридический адрес

Событие (Event)

Параметр	Тип данных	Размер поля	Описание
Id	int		Id события
Date	DateTime		Дата и время начала события в формате «YYYY-MM-dd**T**HH:mm:ss» (Например, 2014-07-31**T**19:00:00)
Duration	int		Длительность события в минутах
VenueId	int		Id площадки, на которой проходит событие
Venue	Venue		Площадка, на которой проходит событие
VenueHallId	int		Id зала площадки, в котором проходит событие
VenueHall	VenueHall		Зал площадки, в котором проходит событие
ActionId	int		Id мероприятия
Action	Action		Мероприятие
ProviderId	int		Id поставщика билетов
Provider	Provider		Поставщик билетов
MainTariffId	int		Id тарифа билетов по умолчанию
TicketCount	int		Количество свободных билетов
TicketType	int		Тип билетов события: 0 – нет билетов, 1 – билет с местом, 2 – входной билет, 3 – содержатся и входные, и билеты с местом.
MinPrice	decimal	(18,2)	Минимальная стоимость билета
MaxPrice	decimal	(18,2)	Максимальная стоимость билета
MinCartPrice	decimal	(18,2)	Минимальная сумма билетов в корзине
MaxCartPrice	decimal	(18,2)	Максимальная сумма билетов в корзине
MinCartTickets	int		Минимальное количество билетов в корзине
ReturnTicketPart	decimal	(18,2)	Какая часть от цены билета в %, будет возвращена клиенту при возврате билета. Например, значение 10 значит, что при цене билета 100р клиенту вернётся 10р и взимается штраф 90р.
SellOpened	bool		Открыты ли продажи по данному событию

SvgFileId	string	22	Id файла зала
AllowEtickets	bool		Возможна ли продажа электронного билета
AllowTickets	bool		Возможна ли продажа обычного билета
Sectors	Sector[]		Список секторов
Restrictions	Attribute[]		Список ограничений для посещения события. (Name - тип ограничения, Value – текст ограничения).

Сектор события (EventSector)

Параметр	Тип данных	Размер поля	Описание
Id	int		Id категории
Type	tinyint (*enum*)		Тип сектора: 1 – сектор с местами, 2 – сектор без мест (входной)
Name	string	100	Наименование сектора
SvgFileId	string	22	Id файла сектора
Count	int		Количество билетов в секторе
MinPrice	decimal	(18,2)	Минимальная стоимость билета в секторе
MaxPrice	decimal	(18,2)	Максимальная стоимость билета в секторе
Integrations	Integration[]		Интеграции с другими системами. В данных объектах есть название системы и ID события в ней.

Мероприятие (Action)

Параметр	Тип данных	Размер поля	Описание
Id	int		Id мероприятия
Title	string	200	Наименование мероприятия
Announcement	string	400	Анонс
Description	Description		Описание
AgeGroup	tinyint (*nullable*)		Возрастная категория (0+, 4+, 6+, 10+, 12+, 16+, 18+). Возможные значения: null, 0, 4, 6, 10, 12, 16, 18 (null – категорию не задали)
CategoryId	int		Id категории мероприятия
Category	Category		Категория мероприятия
SubCategoryId	int		Id подкатегории мероприятия
SubCategory	Category		Подкатегория мероприятия
PosterId	string	22	Id главного файла с постером
PosterIds	string[]		Список Id файлов с изображениями по мероприятию. Главный постер включен в список.
IsPushkinsCard	bool		Возможна ли оплата заказа по Пушкинской карте
NearestEvent	Event		Ближайшее событие
Integrations	Integration[]		Интеграции с другими системами. В данных объектах есть название системы и ID мероприятия в ней.
Restrictions	Attribute[]		Список ограничений для посещения мероприятия. (Name - тип ограничения, Value – текст ограничения).

Описание мероприятия по умолчанию не будет в выборке. Его можно получить через `$expand=Description`, при этом само описание будет в поле `Description.Value`. Например, так:
`/api/Actions(243)?$expand=Description`

Категория (Category)

Параметр	Тип данных	Размер поля	Описание

Id	int		Id категории
Title	string	100	Наименование категории
Description	string	300	Описание категории
IsSubcategory	bool		Является ли данная категория подкатегорией
ParentId	int		Id родительской категории (null – если данная категория не входит ни в какую другую категорию)
Parent	Category		Родительская категория
Childs	Category[]		Список вложенных категорий

Тариф билета (TicketTariff)

Параметр	Тип данных	Размер поля	Описание
Id	int		Id тарифа
Name	string	150	Наименование тарифа
Sort	int		Поле для сортировки тарифов
Price	decimal	(18,2)	Стоимость тарифа по номиналу
ServicePrice	decimal	(18,2)	Стоимость сервисного сбора с билета по данному тарифу

Билет с местом (Ticket)

Параметр	Тип данных	Размер поля	Описание
Id	int		Id билета
Price	decimal	(18,2)	Стоимость билета
ServicePrice	decimal	(18,2)	Сервисный сбор с билета
SectorId	int		Id сектора
Sector	string	200	Наименование сектора
PlaceId	int		Id места
Loge	string	50	Ложа
Row	string	50	Ряд
Seat	string	50	Место
Tariffs	Tariff[]		Список тарифов билета

Билет в корзине (CartTicket)

Параметр	Тип данных	Размер поля	Описание
Id	int		Id билета
Price	decimal	(18,2)	Стоимость билета
ReturnPrice	decimal	(18,2)	Стоимость билета, возвращенная клиенту (если билет в статусе «возвращён»)
ServicePrice	decimal	(18,2)	Сервисный сбор с билета
SectorId	int		Id сектора
Sector	string	200	Наименование сектора
PlaceId	Int (*nullable*)		Id места. Если билет во входной сектор, то значение отсутствует (NULL)
Loge	string	50	Ложа
Row	string	50	Ряд
Seat	string	50	Место
OrderedTariffId	int		Id тарифа, по которому забронирован билет
Barcode	string	12	Штрихкод электронного билета. Присутствует у объекта Ticket только если билет электронный, и бронь оплачена.
EventId	int		Id события
Event	Event		Событие (в выборке только при условии наличия в $expand)

Tariffs	Tariff[]	Список тарифов билета
PromoActionId	int	Id примененной промоакции. Если промоакции в корзине нет – то null.
PromoAction	PromoAction	Примененная промоакция (в выборке только при условии наличия в $expand)
Attributes	Attribute[]	Список атрибутов (в выборке только при условии наличия в $expand)

Бронь (Order)

Параметр	Тип данных	Размер поля	Описание
Id	int		Id брони
InputTimeUTC	DateTime		Время создания брони
CancelTimeUTC	DateTime (*nullable*)		Время окончания брони (null – если времени окончания нет). При наступлении этого времени бронь автоматически отменится.
ClientName	string	200	Наименование клиента
Email	string	200	E-mail клиента
PhoneNumber	string	200	Телефон клиента
State	tinyint (*enum*)		Состояние брони (1 – Принята, 2 – Оплачена, 3 – В обработке (когда есть доставка), 4 – Исполнена, 5 – Отменена)
PaymentType	tinyint (*enum*)		Тип платежа (1 – Наличными, 2 – Банковской картой, 3 – Электронными деньгами)
DeliveryType	tinyint (*enum*)		Тип доставки (1 – Самовыкуп на кассе, 2 – Электронный билет, 3 – Доставка курьером)
TicketCount	tinyint		Кол-во билетов в броне
TicketsPrice	decimal	(18,2)	Стоимость билетов в броне
ServicePrice	decimal	(18,2)	Сервисный сбор с билетов в броне
DeliveryPrice	decimal	(18,2)	Стоимость доставки
FullPrice	decimal	(18,2)	Полная стоимость брони
Comment	string	300	Комментарий к броне
Tickets	CartTicket[]		Список билетов
Delivery	Delivery		Доставка
Attributes	Attribute[]		Список атрибутов (в выборке только при условии наличия в $expand)
UtmTags	UtmTagsInfo		Информация о рекламной компании

Доставка (Delivery)

Параметр	Тип данных	Размер поля	Описание
DeliverFromUTC	DateTime		Желаемая дата доставки (строка в формате YYYY-MM-dd HH:mm:ss)
DeliverToUTC	DateTime		Желаемая дата доставки (строка в формате YYYY-MM-dd HH:mm:ss)
TownId	int		Id города доставки
Street	string	200	Улица, проспект, переулок и т.п. Вводится не только наименование, но и само сокращение «ул.», «пр-кт» или что-то подобное. Например, «ул. Ленина»
House	string	50	Номер дома, включая литеру и/или корпус, если таковые имеются
Flat	string	50	Номер квартиры

Платёж (OrderPayment)

Параметр	Тип данных	Размер поля	Описание
PaymentNumber	string	36	Id платежа в эквайринге
Amount	decimal	(18,2)	Сумма платежа. Платежей может быть несколько, поэтому эта сумма не обязательно должна равняться стоимости брони.
CurrencyCode	smallint (*nullable*)		Код валюты по стандарту ISO 4217. По умолчанию 643.
TransactionTime	DateTimeOffset (*nullable*)		Время совершения конкретного платежа (null – если времени окончания нет, тогда будет присвоено текущее время).
TerminalId	string	36	Идентификатор терминала в эквайринге
MerchantId	string	36	Идентификатор торговой точки в эквайринге
MerchantOrderId	string	36	Идентификатор заказа в эквайринге
MerchantSessionId	string	36	Идентификатор сессии в эквайринге
CardMask	string	36	Маскированный номер карты клиента
CardUid	string	36	Виртуальный номер карты клиента
RRN	string	36	Номер ссылки
AuthCode	string	36	Код авторизации
RspCode	int (*nullable*)		Код ответа платежного приложения
RspMessage	string	1000	Текстовое сообщение с результатом работы платежного приложения
IsPushkinsCard	bool (*nullable*)		Оплачено пушкинской картой

Информация о рекламной компании (UtmTagsInfo)

Параметр	Тип данных	Размер поля	Описание
UtmSource	string	200	Источник трафика (Google, Яндекс, Email, Blog)
UtmMedium	string	200	Тип трафика (PPC, КМС, ретаргетинг, рассылка, пост, баннер)
UtmCampaign	string	200	Название рекламной кампании или рекламного объявления
UtmContent	string	200	Содержание, помогает различать объявления, содержащие одинаковые ключевые слова, может быть полезно для теста объявлений
UtmTerm	string	200	Ключевое слово
UtmReferrer	string	200	Источник перехода

Описание (Description)

Параметр	Тип данных	Размер поля	Описание
Value	string	4000	Описание объекта

Корзина (Cart)

Параметр	Тип данных	Размер поля	Описание
UserSession	string		Сессия пользователя сайта
Ttl	int		Время в секундах до отмены корзины (все билеты в корзине вернутся в продажу, когда параметр станет равным нулю)
Count	int		Количество билетов в корзине
Sum	decimal	(18,2)	Стоимость билетов в корзине (берется полная стоимость с учетом сбора с каждого билета)

OriginalSum	decimal	(18,2)	Стоимость билетов в корзине до применения промоакции (оригинальная стоимость по выбранному тарифу). Если промоакции в корзине нет – то null.
AppliedPromoAction	bool		Применена ли промоакция
PromoActionId	int		Id примененной промоакции. Если промоакции в корзине нет – то null.
Tickets	CartTicket[]		Список билетов в корзине

Промоакция (PromoAction)

Параметр	Тип данных	Размер поля	Описание
PromoId	int		Id промоакции
Name	string	200	Название промоакции
Description	string	200	Описание промоакции
Permanent	bool		Промоакция постоянна. Если промоакция постоянна, то значения **StartTime**, **EndTime** отсутствуют (NULL)
StartTime	DateTime (*nullable*)		Дата начала действия промоакции
EndTime	DateTime (*nullable*)		Дата окончания действия промоакции
WholeDay	bool		Промоакция действует весь день. В случае, если промоакция действует весь день, то значения **StartTimeOfDay**, **EndTimeOfDay** отсутствуют (NULL)
StartTimeOfDay	DateTime (*nullable*)		Ложа
EndTimeOfDay	DateTime (*nullable*)		Ряд
WeekDays	DayOfWeek (*enum*)		Дни недели, в которые действует промоакция
Rules	PromoActionRule[]		Правила, по которым рассчитывается скидка промоакции

Правило промоакции (PromoActionRule)

Параметр	Тип данных	Размер поля	Описание
RuleType	PromoRuleTypes (*enum*)		Тип условия, при котором применяется скидка: `ByTicketsCount` – по количеству билетов в корзине, `ByTicketsPrice` – по сумме билетов в корзине, `ByPromoCode` – по промокоду
PromoCodeType	PromoCodeRuleTypes (*enum*)		Тип условия по промокоду, при котором применяется скидка. Если условие задано не по промокоду - значение отсутствует (NULL): `ByTicketsCount` – по количеству билетов в корзине и промокоду, `ByTicketsPrice` – по сумме билетов в корзине и промокоду, `None` – только по промокоду
DiscountType	PromoDiscountTypes		Тип применяемой скидки:

	(*enum*)		`Percent` – скидка в процентах, `Amount` – скидка в рублях
Discount	decimal	(18,2)	Размер скидки
From	decimal (*nullable*)	(18,2)	В зависимости от типа условия – нижний диапазон количества билетов (`ByTicketsCount`) или суммы в корзине (`ByTicketsPrice`). Если условие задано по промокоду – см. PromoCodeType.
To	decimal (*nullable*)	(18,2)	В зависимости от типа условия – верхний диапазон количества билетов (`ByTicketsCount`) или суммы в корзине (`ByTicketsPrice`). Если условие задано по промокоду – см. PromoCodeType.

Атрибут (Attribute)

Параметр	Тип данных	Размер поля	Описание
Name	string	50	Название атрибута. В пределах объекта должно уникально.
Value	string	200	Значение атрибута.

Информация о событиях в городе/площадке (EventsInfo)

Параметр	Тип данных	Размер поля	Описание
ActionsCount	int		Количество актуальных мероприятий
EventsCount	int		Количество актуальных событий

Информация об интеграции с внешней системой (Integration)

Параметр	Тип данных	Размер поля	Описание
Name	string	50	Название внешней системы. На данный момент поддерживаются значения: • `ProCulture` – идентификатор мероприятия в системе PRO.культура.рф.
ExternalId	int		ID объекта во внешней системе