Глава 17. Механизмы интернет-сервисов

17.1. Web-сервисы

17.1.1. Общая информация

Механизм Web-сервисов в системе «1С:Предприятие» является средством поддержки сервисно-ориентированной архитектуры (Service-Oriented Architecture, SOA).

Сервисно-ориентированная архитектура представляет собой прикладную архитектуру, в которой все функции определены как независимые сервисы с вызываемыми интерфейсами. Обращение к этим сервисам в определенной последовательности позволяет реализовать тот или иной бизнес-процесс.

Сервисно-ориентированная архитектура предлагает новый подход к созданию распределенных информационных систем, в которых программные ресурсы рассматриваются как сервисы, предоставляемые по сети. Такой подход позволяет обеспечить быструю консолидацию распределенных компонентов (сервисов) в единое решение для поддержки определенных бизнес-процессов.

Механизм Web-сервисов позволяет использовать систему «1С:Предприятие» как набор сервисов в сложных распределенных и гетерогенных системах, а также позволяет интегрировать ее с другими промышленными системами с использованием сервисно-ориентированной архитектуры.

Конфигурация системы «1С:Предприятие» может экспортировать свою функциональность через Web-сервисы. Определения Web-сервисов задаются в дереве конфигурации и становятся доступны произвольным информационным системам благодаря публикации их на веб-сервере.

Кроме этого, система «1С:Предприятие» может обращаться к Web-сервисам сторонних производителей как через статические ссылки, определенные в дереве конфигурации, так и с помощью динамических ссылок, создаваемых средствами встроенного языка.

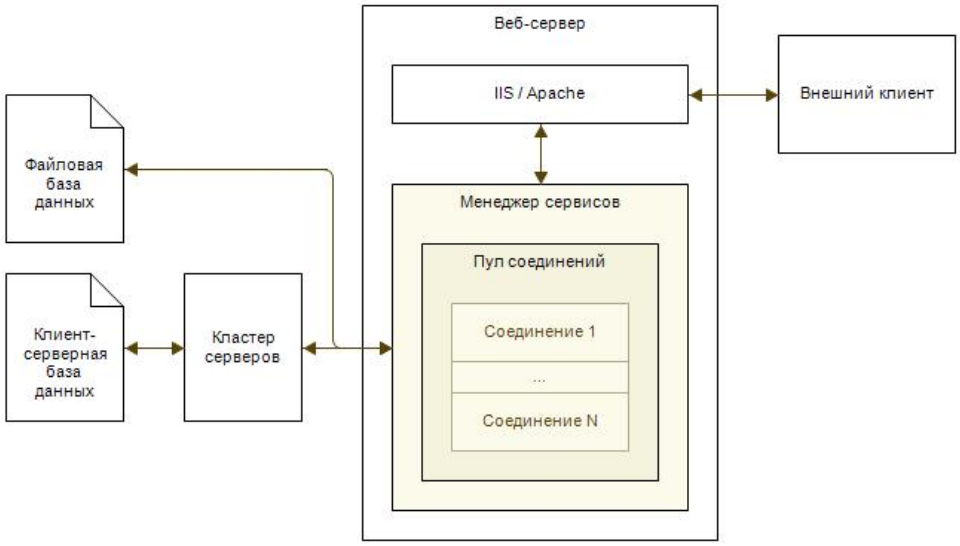

Рис. 458. Web-сервисы

В основе сервисной архитектуры системы «1С:Предприятие 8» находится менеджер сервисов. Менеджер сервисов выполняет следующие функции:

- управление пулом соединений с информационными базами;
- поддержка WSDL описания сервиса;
- реализация протокола SOAP, сериализация сообщений, вызов соответствующего сервиса.

Менеджер сервисов выполняется в процессе сервисного хоста, который выполняет функцию приема/передачи сообщений из/в менеджер сервисов. В качестве сервисного хоста может использоваться веб-сервер IIS или Apache.

Менеджер сервисов содержит в себе пул соединений, через которые идет взаимодействие с базами данных системы «1С:Предприятие».

Механизм Web-сервисов, реализованный в системе «1С:Предприятие», поддерживает следующие стандарты:

- SOAP 1.1,
- SOAP 1.2,
- WSDL 1.1,
- WS-I Basic Profile 1.1,
- HTTP 1.1,
- TLS 1.x (TLS 1.1 и 1.2 поддерживаются, если не требуется передача клиентского сертификата на сервер), включая криптографические алгоритмы, соответствующие ГОСТ Р 34.10-94, Р 34.10-2001, Р 34.10-2012, Р 34.11-94, Р 34.11-2012 и 28147-89;
- MTOM;
- Аутентификация: Basic, NTLM/Negotiate.

Механизм Web-сервисов, реализованный в системе «1С:Предприятие», не поддерживает:

- Механизм WS-Policy;
- Механизм WS-Addressing;
- Протокол WS-Security;
- Задание уже используемого XML пространства имен в качестве целевого пространства имен (targetNamespace).

Для получения доступа к Web-сервису необходимо использовать адрес, который формируется следующим образом: http://host/base/ws/ИмяWebСервиса или http://host/base/ws/АдресWebСервиса. Более подробно рассмотрим составные части адреса:

- http://host/base – обычный URL, по которому выполняется доступ, например, к информационной базе с помощью веб-клиента. При наличии разделителей, не поддерживается указание значений разделителей с помощью параметра Z командной строки запуска клиентского приложения.
- ws – признак того, что выполняется обращение к Web-сервису (в отличие от hs, который определяет доступ к HTTP-сервису, см. здесь).
- ИмяWebСервиса – имя Web-сервиса. Задается в свойстве объекта Web-сервис.
- АдресWebСервиса – описывает альтернативное имя для доступа к Web-сервису. Задается в свойстве Имя файла публикации объекта Web-сервис. Может быть изменено при публикации Web-сервиса.
- Обращения по имени и адресу Web-сервиса являются равносильными.

17.1.2. Предоставление функциональности через Web-сервисы

Для того чтобы функциональность системы «1С:Предприятие» стала доступна внешним потребителям Web-сервисов, нужно выполнить следующие действия:

- создать в конфигурации необходимое количество Web-сервисов,
- опубликовать Web-сервисы с помощью специального инструмента конфигуратора.

Описание процедуры публикации Web-сервисов см. здесь. Описание использования Reverse Proxy для доступа к «1С:Предприятию» см. здесь.

Создание Web-сервиса заключается:

- в добавлении в дерево метаданных объекта конфигурации Web-сервис,
- описании операций, которые может выполнять создаваемый Web-сервис,
- описании параметров операций Web-сервиса.

Объект конфигурации Web-сервис содержит модуль, в котором создаются процедуры на встроенном языке, выполняемые при вызове тех или иных операций Web-сервиса. Типы параметров операций Web-сервиса описываются с помощью типов XDTO и могут представлять собой либо значения XDTO, либо объекты XDTO.

Вызов Web-сервиса происходит следующим образом:

- из пула соединений выбирается подходящее соединение с информационной базой; при отсутствии необходимого соединения соединение создается;
- создается новый сеанс и для созданного сеанса вызывается событие `УстановкаПараметровСеанса` (в модуле сеанса);
- выполняется вызов затребованного метода Web-сервиса, при этом происходит вызов обработчика `УстановкаПараметровСеанса()` (в модуле сеанса) каждый раз, когда происходит обращение к неинициализированному параметру сеанса.

СОВЕТ. Не рекомендуется выполнять ресурсоемкие операции в обработчике события `УстановкаПараметровСеанса`.

Событие `УстановкаПараметровСеанса` модуля сеанса вызывается на сервере в привилегированном режиме. Модуль вызванного сервиса исполняется на сервере в обычном режиме.

Модуль сеанса (см. здесь) служит для инициализации параметров сеанса и выполнения некоторого набора команд при вызове любого Web-сервиса системы «1С:Предприятие».

17.1.3. Пример реализации Web-сервиса

Например, требуется создать Web-сервис системы «1С:Предприятие», который должен по переданному номеру расходной накладной возвращать состав ее табличной части. Аналогичный пример с помощью HTTP-сервисов см. здесь.

Для описания возвращаемого значения создадим пакет XDTO `ДанныеРасходнойНакладной` с пространством имен http://www.MyCompany.ru/shipment, содержащий три типа объектов XDTO:

- `Номенклатура` – для передачи данных элемента справочника `Номенклатура`. Этот тип объекта XDTO будет содержать следующие свойства:
  - `Наименование` – тип `string` из пространства имен http://www.w3.org/2001/XMLSchema;
  - `ПолноеНаименование` – тип `string` из пространства имен http://www.w3.org/2001/XMLSchema;
  - `ШтрихКод` – тип `string` из пространства имен http://www.w3.org/2001/XMLSchema;
  - `ЗакупочнаяЦена` – тип `int` из пространства имен http://www.w3.org/2001/XMLSchema.
- `СтрокаРасходнойНакладной` – для передачи данных одной строки расходной накладной. Этот тип объекта XDTO будет содержать следующие свойства:
  - `Номенклатура` – тип `Номенклатура` из пространства имен http://www.MyCompany.ru/shipment; представляет собой ссылку на объект XDTO, который мы определили выше;

- Количество – тип int из пространства имен http://www.w3.org/2001/XMLSchema;
- Цена – тип int из пространства имен http://www.w3.org/2001/XMLSchema;
- Сумма – тип int из пространства имен http://www.w3.org/2001/XMLSchema.

- РасходнаяНакладная – для передачи данных всех строк расходной накладной. Этот тип объекта XDTO будет содержать единственное свойство:
  - Состав – тип СтрокаРасходнойНакладной из пространства имен http://www.MyCompany.ru/shipment. Представляет собой ссылку на объект XDTO, который мы определили выше. Для того чтобы это свойство могло содержать неограниченное множество значений, необходимо установить его свойство Верхняя граница в значение -1.

После того как необходимые типы XDTO созданы, следует добавить в конфигурацию новый Web-сервис ДанныеРасходнойНакладной со следующими значениями свойств:

- URI Пространства имен – http://www.MyCompany.ru/shipment;
- Пакеты XDTO – ДанныеРасходнойНакладной;
- Имя файла публикации – shipment.1cws.

У созданного Web-сервиса следует определить операцию Получить со следующими значениями свойств:

- Тип возвращаемого значения – РасходнаяНакладная из пространства имен http://www.MyCompany.ru/shipment;
- Возможно пустое значение – установлен;
- Имя процедуры – Получить.

У операции Получить следует определить параметр НомерДокумента со следующими значениями свойств:

- Тип значения – тип string из пространства имен http://www.w3.org/2001/XMLSchema;
- Направление передачи – Входной.

После этого следует открыть модуль созданного Web-сервиса и разместить в этом модуле функцию Получить(), которая будет выполняться при вызове данного Web-сервиса.

```
ФункцияПолучить(НомерДокумента)Экспорт
^//Получитьобъектрасходнойнакладнойпопереданномуномеру
^ДокументСсылка=Документы.РасходнаяНакладная.НайтиПоНомеру(НомерДокумента,ТекущаяДата());
^ЕслиДокументСсылка.Пустая()Тогда
^^ВозвратНеопределено;
^КонецЕсли;
^Документ=ДокументСсылка.ПолучитьОбъект();
^//Получитьтипыобъектовxdto
^НоменклатураТип=ФабрикаXDTO.Тип("http://www.MyCompany.ru/shipment","Номенклатура");
^РасходнаяНакладнаяТип=ФабрикаXDTO.Тип("http://www.MyCompany.ru/shipment","РасходнаяНакладная");
^СтрокаРасходнойНакладнойТип=ФабрикаXDTO.Тип("http://www.MyCompany.ru/shipment","СтрокаРасходнойНакладной");
^//СоздатьобъектXDTOрасходнойнакладной
^РасходнаяНакладная=ФабрикаXDTO.Создать(РасходнаяНакладнаяТип);
^ДляКаждогоСтрокаДокументаИзДокумент.СоставЦикл
^^//СоздатьобъектыXDTOстрокирасходнойнакладной
^^//иноменклатуры
^^СтрокаРасходнойНакладной=ФабрикаXDTO.Создать(СтрокаРасходнойНакладнойТип);
^^Номенклатура=ФабрикаXDTO.Создать(НоменклатураТип);
^^//Заполнитьсвойстваноменклатуры
^^Номенклатура.Наименование=СтрокаДокумента.Номенклатура.Наименование;
^^Номенклатура.ПолноеНаименование=СтрокаДокумента.Номенклатура.ПолноеНаименование;
^^Номенклатура.ШтрихКод=СтрокаДокумента.Номенклатура.ШтрихКод;
^^Номенклатура.ЗакупочнаяЦена=СтрокаДокумента.Номенклатура.ЗакупочнаяЦена;
^^//Заполнитьсвойствастрокирасходнойнакладной
^^СтрокаРасходнойНакладной.Номенклатура=Номенклатура;
^^СтрокаРасходнойНакладной.Количество=СтрокаДокумента.Количество;
^^СтрокаРасходнойНакладной.Цена=СтрокаДокумента.Цена;
^^СтрокаРасходнойНакладной.Сумма=СтрокаДокумента.Сумма;
^^//Добавитьстрокурасходнойнакладной
^^РасходнаяНакладная.Состав.Добавить(СтрокаРасходнойНакладной);
^КонецЦикла;
^//Вернутьрасходнуюнакладную
^ВозвратРасходнаяНакладная;
КонецФункции
```

В заключение следует опубликовать созданный Web-сервис на веб-сервере, например http://www.MyCompany.ru, в каталоге shipment.

17.1.4. Работа с веб-сервисами сторонних поставщиков

17.1.4.1. Общая информация

Система «1С:Предприятие» может использовать веб-сервисы, предоставляемые другими поставщиками, несколькими способами:

- с помощью статических ссылок, создаваемых в дереве конфигурации;
- с помощью динамических ссылок, создаваемых средствами встроенного языка;
- комбинацией предыдущих способов.

Преимущество использования статических ссылок заключается в большей скорости работы, т. к. описание веб-сервиса поставщика получается один раз, при создании ссылки. В дальнейшем при обращении к данному веб-сервису используется существующее описание веб-сервиса.

При использовании динамических ссылок описание веб-сервиса поставщика будет получаться системой «1С:Предприятие» каждый раз при вызове веб-сервиса, что, естественно, будет замедлять работу с данным веб-сервисом. Однако преимуществом такого подхода является возможность получения актуального описания веб-сервиса поставщика. При использовании же статических ссылок для получения актуального описания веб-сервиса следует выполнить повторный импорт WSDL-описания средствами конфигуратора и сохранение измененной конфигурации.

При эксплуатации прикладных решений может возникать ситуация, когда один и тот же веб-сервис предоставляется по разным адресам (URL), однако имеет абсолютно одинаковое описание (WSDL). В этом случае возникает потребность загрузить описание веб-сервиса в конфигурацию (создать объект в дереве объектов конфигурации), но во время использования указать конкретный адрес, по которому расположен веб-сервис. Комбинированный способ позволяет работать таким образом. В качестве примера можно рассмотреть следующую ситуацию: есть тиражируемый веб-сервис, выполняющий некоторую функцию. Прикладное решение, написанное на «1С:Предприятии», пользуется услугами данного сервиса, при этом адрес сервиса может быть различным (сервис тиражируемый), а описание – фиксированное. Тогда в прикладное решение может загрузить описание веб-сервиса, а в настройках прикладного решения предусмотреть ввод адреса конкретного экземпляра сервиса, который (адрес) и будет использоваться при работе.

Также доступно и другой способ: используется динамическая ссылка, но адрес расположения веб-сервиса получается не из файла описания (WSDL), а непосредственно указывается при создании объекта.

При попытке загрузить описание Web-сервиса в конфигураторе (создание статической ссылки) или при использовании динамической ссылки (при помощи объекта `WSОпределения`), система выполняет проверку загружаемого описания Web-сервиса (WSDL). Если в описании Web-сервиса присутствует ошибка (с «точки зрения» системы «1С:Предприятия»), то описание не будет загружено и будет сгенерировано исключение. В тексте исключения будет находиться подробная диагностика причин отказа в загрузке. Ошибки WSDL располагаются в порядке их обнаружения. Каждая ошибка WSDL содержит запись с детальным описанием следующего вида:

```
<Типэлементасошибкой>.<Имя>
^[<Типэлементасошибкой>.<Имя>[…]]
^^<Описаниеошибки>
^^<Типэлементасошибкой>
^^<Имя>
```

В этом описании:

- Тип элемента с ошибкой – тип элемента WSDL. Чаще всего соответствует имени тега в xml-файле, описывающем Web-сервис.
- Описание ошибки – описание конкретной ошибки.
- Имя – имя объекта WSDL, в котором произошла ошибка.

В нижеследующей таблице приведены ошибки, которые могут возникнуть при проверке описания Web-сервиса, и рекомендации по их (ошибок) исправлению:

Описание ошибки	**Рекомендации по исправлению**
Тип не найден	Проверить корректность пространств имен и имен типов и элементов
Сообщение не найдено	Проверить корректность пространств имен и имен сообщений
Привязка не найдена	Проверить корректность пространств имен и имен привязок
Тип порта не найден	Проверить корректность пространств имен и имен типов портов
Абстрактная операция не найдена	Проверить корректность имени абстрактной операции
Содержимое операции не найдено	Необходимо в операции указать содержимое. Это может быть вход или выход операции
Описание ошибки не найдено	Необходимо указать имя описания ошибки
Абстрактная операция не задана	Необходимо указать абстрактную операцию
Имя не задано	Необходимо указать имя объекта WSDL согласно схемы: http://schemas.xmlsoap.org/wsdl/2003-02-11.xsd
Вход операции не соответствует входу абстрактной операции	Необходимо, чтобы операция и абстрактная операция либо имели вход (обе), либо не имели его (обе). Нельзя, чтобы абстрактная операция имела вход, а операция нет и наоборот.
Выход операции не соответствует выходу абстрактной операции	Аналогично сообщению про вход операции
Тип порта не задан	Необходимо указать тип порта, проверить пространства имен
Объект не уникален	Необходимо проверить уникальность объекта по пространству имен и имени
Расположение не задано	Необходимо указать расположение Правило R2007 – для элемента import Правило R2711 – для элемента port
Пространства имен отличаются	Необходимо проверить и исправить отличия пространств имен импорта и пространства имен импортируемой схемы
Тип части сообщения не задан	Для части сообщения необходимо задать тип (тип или элемент, в зависимости от стиля WSDL), проверить пространства имен
Сообщение не задано	Для объекта WSDL необходимо указать или исправить имя сообщения, проверить его пространство имен
Абстрактной операции требуется вход	Необходимо указать вход для абстрактной операции
Абстрактной операции требуется отсутствие входа	Абстрактная операция не должна иметь вход
Абстрактной операции требуется выход	Необходимо указать выход для абстрактной операции

Абстрактной операции требуется отсутствие выхода	Абстрактная операция не должна иметь выход
Ошибка абстрактной операции не задана	Необходимо указать или проверить корректность имени абстрактной ошибки, пространства имен
Привязка не задана	Привязка порта не задана, некорректное имя привязки, некорректное пространство имен
Абстрактное описание операции не задано	Абстрактное описание операции не задано, некорректно указано имя или пространство имен
Пространство имен операции в rpc-literal стиле должно быть задано	Необходимо указать элемент namespace для операции. Правило R2717
Пространство имен операции в document-literal стиле не должно быть задано	Необходимо удалить элемент namespace для операции. Правило R2216
В rpc-literal стиле часть сообщения должна ссылаться на определение типа в схеме	Для указания типа части необходимо у соответствующего объекта message в частях сообщений использовать тип, а не элемент (тег type, а не element). Правило R2203
В document-literal стиле часть сообщения должна ссылаться на объявление элемента в схеме	Для указания типа части необходимо у соответствующего объекта message в частях сообщений использовать элемент, а не тип (тег element, а не type). Правило R2204
Неизвестный стиль привязки	Необходимо указать стиль привязки и проверить правильность его написания. Спецификация: http://www.w3.org/TR/wsdl#_bindings. Правила R2705, R2706
Транспорт в привязке должен быть задан как http://schemas.xmlsoap.org/soap/http	Необходимо проверить, чтобы транспорт был именно http://schemas.xmlsoap.org/soap/http. Правило R2702
Часть сообщения не задана	Необходимо для описания ошибку задать часть сообщения. Правило R2205
В схеме ошибка заголовка должна быть описана как объявление элемента	Необходимо в части сообщения, которая используется для описания ошибки использовать тип заданный как элемент (использовать тег element). Правило R2205
В схеме заголовок должен быть описан как объявление элемента	Необходимо в части сообщения, которая используется для описания заголовка использовать тип заданный как элемент (использовать тег element). Правило R2205
Схема должна содержать атрибут targetNamespace	Необходимо для схемы добавить атрибут targetNamespace и заполнить его. Это требование связано с особенностями реализации платформы
Ошибка загрузки импортируемой схемы	Проверить корректность пространств имен, расположения импортируемой схемы

При указании в таблице выражения вида Правило RXXXX понимается следующее – в спецификации Basic Profile Version 1.1 (http://www.ws-i.org/profiles/basicprofile-1.1-2004-08-24.html, на английском языке) необходимо найти правило RXXXX.

17.1.4.2. Пример использования статической WS-ссылки

В качестве примера использования веб-сервисов стороннего поставщика рассмотрим обращение к Web-сервису, который ранее был создан в примере (см. здесь).

Прежде всего, следует добавить в дерево конфигурации новый объект конфигурации WS-ссылка с именем ДанныеРасходнойНакладной, ссылающийся на опубликованный сервис. Для этого следует выполнить импорт WSDL-описания опубликованного сервиса и в качестве URL указать http://www.MyCompany.ru/shipment/ws/Shipment.1cws?wsdl. Описание импорта WSDL-описания см. здесь.

После этого, например, в модуле приходной накладной, можно создать процедуру, приведенную ниже. Она заполняет табличную часть документа данными расходной накладной поставщика, полученными с помощью веб-сервиса поставщика.

```
ПроцедураПолучитьДанныеРасходнойНакладной(НомерНакладнойПоставщика)
^//СоздатьWS-проксинаосновaнииссылки
^Прокси=WSСсылки.ДанныеРасходнойНакладной.-
СоздатьWSПрокси("http://www.MyCompany.ru/shipment","ДанныеРасходнойНакладной","ДанныеРасходнойНакладнойSoap");
^ДанныеНакладной=Прокси.Получить();
^ЕслиДанныеНакладной=НеопределеноТогда
^^Возврат;
^КонецЕсли;
^//Заполнитьприходнуюнакладнуюполученнымиданными
^ДляКаждогоСтрокаНаклИзДанныеНакладной.СоставЦикл
^^НоваяСтрока=ДокументОбъект.Состав.Добавить();
^^НоваяСтрока.Количество=СтрокаНакл.Количество;
^^НоваяСтрока.Цена=СтрокаНакл.Цена;
^^НоваяСтрока.Сумма=СтрокаНакл.Сумма;
^^//Найтиэлементноменклатурыпопереданнымданным
^^//(например,поштрихкоду)
^^НоваяСтрока.Номенклатура=Справочники.Номенклатура.НайтиПоРеквизиту("ШтрихКод",СтрокаНакл.Номенклатура.ШтрихКод);
^КонецЦикла;
КонецПроцедуры
```

Если адрес реального расположения сервиса отличается от адреса, который использовался во время загрузки описания веб-сервиса в конфигурацию, то новый адрес необходимо явно указать при создании объекта WSПрокси:

```
//СоздатьWS-проксинаосновании ссылки
Прокси=WSСсылки.ДанныеРасходнойНакладной.-
СоздатьWSПрокси("http://www.MyCompany.ru/shipment","ДанныеРасходнойНакладной","ДанныеРасходнойНакладнойSoap",,,,"http://
```

17.1.4.3. Пример использования динамической WS-ссылки

Использование динамической ссылки отличается от использования статической ссылки только способом создания WS-прокси и отсутствием необходимости создавать WS-ссылку в дереве конфигурации.

Если провести сравнение с примером, представленным в предыдущем разделе, то, в отличие от создания прокси на основе статической ссылки, при использовании динамической ссылки WS-прокси создается с помощью конструктора следующим образом:

```
//СоздатьWS-проксинаосновании WS-определения
Определение=НовыйWSОпределения("http://www.MyCompany.ru/shipment/ws/Shipment.1cws?wsdl");
Прокси=Новый-
WSПрокси(Определение,"http://www.MyCompany.ru/shipment","ДанныеРасходнойНакладной","ДанныеРасходнойНакладнойSoap");
```

В то время как создание WS-прокси на основе статической ссылки выглядит следующим образом:

```
//СоздатьWS-проксинаосновании ссылки
Прокси=WSСсылки.ДанныеРасходнойНакладной.-
СоздатьWSПрокси("http://www.MyCompany.ru/shipment","ДанныеРасходнойНакладной","ДанныеРасходнойНакладнойSoap");
```

Если адрес реального расположения сервиса отличается от адреса, который указан в WSDL-файле, используемом при создании определения веб-сервисов, новый адрес необходимо явно указать при создании объекта WSПрокси на основании определения веб-сервисов:

```
//СоздатьWS-проксинаосновании WS-определения
Определение=НовыйWSОпределения("http://www.MyCompany.ru/shipment/ws/Shipment.1cws?wsdl");
Прокси=Новый-
WSПрокси(Определение,"http://www.MyCompany.ru/shipment","ДанныеРасходнойНакладной","ДанныеРасходнойНакладнойSoap",,-
,,"http://www.realURL/realPath");
```

17.1.5. Редактирование свойств Web-сервиса

На закладке Основные вводится имя, синоним и комментарий объекта.

На закладке Операции создаются подчиненные объекты Операции, которые, в свою очередь, могут иметь подчиненные объекты Параметры, необходимые для работы с объектами данного типа. Описание подчиненных объектов выполняется в палитре свойств.

17.1.5.1. Свойства «Операции»

Помимо общих свойств объектов конфигурации операция Web-сервиса содержит следующие свойства:

- Тип возвращаемого значения – тип значения, которое возвращает операция Web-сервиса. Может являться типом значения XDTO или типом объекта XDTO.
- Возможно пустое значение – показывает, может ли возвращаемое значение принимать неопределенное значение.
- В транзакции – показывает, будет ли выполняться код модуля Web-сервиса в транзакции или нет. Если свойство установлено, то при вызове Web-сервиса автоматически будет начата транзакция, а при завершении работы Web-сервиса транзакция будет либо зафиксирована, либо произойдет откат транзакции (в зависимости от результатов выполнения). Если свойство не установлено, при начале исполнения модуля Web-сервиса транзакция не будет начата.
- Имя метода – имя экспортируемой процедуры модуля Web-сервиса, которая будет выполнена при вызове данного свойства.

На закладке Подсистемы указывается, к каким подсистемам относятся объекты данного типа.

На закладке Прочее определяются следующие свойства:

- URI пространства имен – содержит URI пространства имен Web-сервиса. Каждый Web-сервис может быть однозначно идентифицирован по своему имени и URI пространству имен, которому он принадлежит. Пространство имен сервиса не должна совпадать с известными пространствами имен, которые уже используются или зарезервированы другими организациями. Рекомендуется в пространство имен сервиса включать фрагмент, уникальный для организации, которая ведет разработку Web-сервиса. Например, для организации с названием Промресурс имеет смысл начинать все пространства имен с префикса, например, http://promresurs.com. Тогда URI пространства имен Web-сервиса будет иметь вид http://promresurs.com/public/services/OurService.
- Пакеты XDTO – перечень пакетов XDTO, типы которых могут использоваться в качестве типов возвращаемого значения операций и типов параметров операций Web-сервиса.
- Имя файла публикации – имя файла описания Web-сервиса, который расположен на веб-сервере.

По кнопке Модуль открывается редактор модуля Web-сервиса.

17.1.5.2. Свойства «Параметр»

Помимо общих свойств объектов конфигурации параметр операции Web-сервиса содержит следующие свойства:

- Тип значения – тип значения параметра операции Web-сервиса. Может являться типом значения XDTO или типом объекта XDTO.
- Возможно пустое значение – показывает, может ли значение параметра операции принимать неопределенное значение.
- Направление передачи – определяет направление передачи данных с помощью данного параметра. Возможные значения:
  - Входной – означает, что параметр используется для передачи данных Web-сервису;
  - Выходной – означает, что параметр используется для получения данных от Web-сервиса;

- Входной-Выходной – означает, что параметр может использоваться как для передачи данных, так и для их получения от Web-сервиса.

17.2. HTTP-сервисы

17.2.1. Стандартный интерфейс OData

17.2.1.1. Общая информация

Стандартный интерфейс OData в «1С:Предприятии» предназначен для получения доступа к данным системы из внешнего приложения без модификации кода прикладного решения (например, если прикладное решение стоит на поддержке). Для получения такого доступа необходимо особым образом опубликовать приложение на веб-сервере и указать, какие объекты конфигурации будут использоваться таким образом (см. здесь).

Для доступа к данным используется протокол OData (http://www.odata.org/, на английском языке) версии 3 (http://www.odata.org/documentation/odata-version-3-0/odata-version-3-0-core-protocol, на английском языке). Поддерживаются следующие форматы представления данных: **atom-xml** и **json**. Для доступа к данным, при публикации, автоматически генерируется стандартный интерфейс OData, который позволяет читать данные «1С:Предприятия», изменять их, создавать новые объекты данных и удалять существующие. Прикладное решение на базе «1С:Предприятия» может выступать как клиентом, так и сервером при работе со стандартным интерфейсом OData. Для работы сервером практически никаких дополнительных действий осуществлять не надо (эта возможность предоставляется автоматически). Для того, чтобы стать клиентом стандартного интерфейса OData, необходимо в прикладном решении реализовать программный слой, который будет использовать данные сервера с использованием стандартных интерфейсов «1С:Предприятия», например объекта `HTTPСоединение`.

В одной системе могут одновременно существовать и SOAP-сервисы (которые реализуются в «1С:Предприятие» в виде Web-сервисов, см. здесь) и стандартный интерфейс OData и HTTP-сервис (см. здесь).

Стандартный интерфейс OData можно использовать для следующих задач:

- Интеграция прикладного решения с различными сайтами (например, на базе Microsoft SharePoint);
- Реализация сторонними средствами дополнительной функциональности прикладного решения без изменения его конфигурации;
- Загрузка данных в прикладное решение и выгрузка данных из него;
- Интеграция прикладного решения с корпоративными системами, возможно даже без дополнительного программирования.

Типичные операции, выполняемые с помощью стандартного интерфейса OData:

- Получение списка объектов системы с установленным отбором;
- Получение данных конкретного объекта системы;
- Создание нового и запись изменений существующего объекта системы;
- Проведение одного документа, старт бизнес-процесса.

17.2.1.2. Общие принципы работы

При работе с протоколом OData используется специальная терминология. При применении протокола используются следующие термины:

- **Сущность** – нечто, обладающее идентичностью. Сущность также обладает набором свойств. Некоторые свойства описывают ее идентичность, комбинация которых определяет ключ сущности. По этому ключу можно получить конкретную сущность.
- **Набор сущностей** – коллекция сущностей определенного типа.
- **Составной тип** – набор свойств, не обладающий идентичностью.
- **Функция** – набор некоторых операций, выполняемых на стороне сервера, возвращающий данные (не обязательно сущность или набор сущностей) и не приводящий к наблюдаемым побочным эффектам (изменениям данных). Функция обязательно связана с сущностью или набором сущностей.
- **Действие** – функция, которая может изменять данные.

Термины протокола некоторым образом соответствуют терминам, принятым в системе «1С:Предприятие»:

- **Сущность** – в качестве сущности может выступать один из 4 групп объектов системы «1С:Предприятие»: объектные типы, наборы записей регистров, записи регистров и строки табличных частей объектных типов. Записи регистров и строки табличных объектных типов (как отдельные сущности) доступны только на чтение.
- Реквизиты объектов «1С:Предприятия» представляются как **свойства** сущностей. В некоторых случаях (например, реквизит объекта конфигурации составного типа) реквизит может быть представлен несколькими свойствами, одно из которых будет **навигационным**. Такое свойство содержит в качестве значения ссылку (URL) на сущность, описывающую объект «1С:Предприятия». Свойство, описывающее тип такого реквизита, называется **диспетчеризационным**. Название такого свойства завершается суффиксом _Type. Более подробное описание диспетчеризационного свойства приведено далее.
- Если реквизит объекта имеет ссылочный тип, то объект, на который ссылается такой реквизит, будет называться **связанной сущностью**.

Обращение к стандартному интерфейсу OData выполняется с помощью HTTP-запроса, выполнено по определенному URL. URL формируется специальным образом и состоит из следующих частей:

1. Адрес информационной базы;
2. Признак обращения к стандартному интерфейсу OData;
3. Имя ресурса, к которому выполняется обращение;
4. Параметры запроса обращения к ресурсу.

Само обращение выполняется с помощью HTTP-запроса определенного вида. При более подробном описании работы со стандартным интерфейсом OData вид запроса будет указываться отдельно.

Рассмотрим части URL более подробно.

Адрес информационной базы

Это обычный URL, по которому выполняется доступ, например, к информационной базе с помощью веб-клиента. Например, http://host/base или http://host.server.zone/data-base. Также необходимо помнить, что при использовании информационной базы, в которой настроено разделение, допустимо использовать только один способ указания значений разделителей, а именно: значения разделители можно указывать только в URL информационной базы. Указание разделителей с помощью параметра Z не поддерживается. Более подробную информация про настройку указания значений разделителей в URL информационной базы можно получить в описании файла default.vrd в книге 1С:Предприятие 8.3. "Руководство администратора".

Признак обращения к стандартному интерфейсу OData

В качестве такого признака выступает последовательность /odata/standard.odata.

Имя ресурса, к которому выполняется обращение

Особым образом сформированный идентификатор ресурса (возможно, с параметром) или предопределенные ресурсы. Например, $metadata, Catalog_Контрагент(guid'value').

Параметры обращения к ресурсу

В качестве параметров обращения выступают параметры в виде, принятом для HTTP-запросов: ?ключ=значение&ключ2=значение2.

При обращении к ресурсу могут использоваться специальные ключевые слова, имеющие специальное назначение:

- $format – указывает, в каком формате необходимо получить данные. Если ключевое слово не указано, данные получаются в формате atom-xml.
  - $format=atom – возвращает данные в формате **atom-xml**.
  - $format=json – возвращает данные в формате **json**. Для указания того, что данные должны возвращаться в формате **json**, можно указать MIME-тип application/json в заголовке Accept HTTP-запроса на получение данных.
- $metadata – указывает, что требуется получить описание стандартного интерфейса OData. Подробнее см. здесь.
- $filter – описывает отбор, применяемый при получении данных. Подробнее см. здесь.
- $select – описывает перечень свойств сущности, которые получаются при обращении к стандартному интерфейсу OData. Подробнее см. здесь.

После того, как сформирован URL необходимого ресурса, следует выполнить HTTP-запрос нужного вида. В зависимости от того, какая операция выполняется, используется соответствующий HTTP-метод:

- Получение данных – метод `GET`;
- Создание объекта – метод `POST`;
- Обновление данных:
  - метод `PATCH` – в этом случае можно указывать только те свойств, которые необходимо обновить;
  - метод `PUT` – в этом случае необходимо указывать все свойства сущности;
- Удаление данных – метод `DELETE`.

В результате выполнения запроса клиентское приложение получает ответ сервера, который кроме кода состояния может содержать различные данные, предоставленные сервером, в виде XML-документа.

В примерах URL, которые используются в данном разделе, выражение guid'value' означает выражение OData, которое описывает конкретное значение GUID (тип `Edm.Guid`).

17.2.1.3. Представление данных

Данные, возвращаемые стандартным интерфейсом OData, могут быть представлены в виде XML-документа или JSON-документа. Это зависит от того, в каком формате запрашиваются данные.

Для различных типов данных используется следующее соответствие между типом «1С:Предприятия» и типом OData:

Тип «1С:Предприятия»	**Тип свойства OData**
`Строка`	`Edm.String`
`Дата`	`Edm.DateTime`
`Число` (целое)	`Edm.Int16`, `Edm.Int32`, `Edm.Int64`
`Число` (дробное)	`Edm.Double`
`Булево`	`Edm.Boolean`
`Ссылка`	`Edm.Guid`
`Перечисление`	`Edm.String`
`ХранилищеЗначения`	Состоит из трех свойств*: 1. Навигационное свойство; 2. <Имя свойства>_Base64Data;

	3. <Имя свойства>_Type.
Ссылочный тип	Два свойства: • Собственно значение ссылки на объект типа Edm.Guid; • Навигационное свойство.
Составные типы	Два свойства типа Edm.String: • Значение фактического типа, выраженное в виде строки; • Имя фактического типа.

В таблице символ * означает, что:

- С помощью навигационного свойства можно получить данные, хранящиеся в реквизите:
  - Для типа Картинка – будет получена собственно картинка с соответствующим значением HTTP-заголовка content-type;
  - Для типа ДвоичныеДанные – будет получен байтовый поток;
  - Для остальных типов – XDTO-сериализованное значение хранимых данных.
- Свойство <Имя свойства>_Base64Data хранит данные, которые могут быть получены с помощью навигационного свойства, только закодированные в Base64. Редактировать данные реквизита типа ХранилищеЗначения можно только с помощью этого свойства.
- Свойство <Имя свойства>_Type описывает тип данных, хранимых в реквизите. Может принимать одно и трех значений:
  - application/octet-stream – двоичные данные;
  - application/xml+xdto – XDTO-сериализованный объект;
  - значение HTTP-заголовка content-type, соответствующего картинке, хранящейся в реквизите, например, image/jpeg для картинки формата JPEG.

Остальные типы не поддерживаются, и при попытке их чтения будет сгенерирована ошибка с кодом 501.

Имена свойств могут оканчиваться на различные суффиксы. Такие свойства имеют специальный смысл. Далее будут более подробно рассмотрены возможные суффиксы:

- Key;
- Type;
- Base64Data;
- _____Presentation.

Key

Свойство с таким суффиксом содержит значение ключа соответствующего ссылочного реквизита объекта (без такого суффикса) или независимого регистра сведений без измерений. Имя с таким суффиксом используется для установки отбора в качестве имени реквизита, по которому выполняется отбор. Например, для установки отбора по ссылочному полю Контрагент, условие будет выглядеть следующим образом: Контрагент_Key=guid'value'. При этом свойство Контрагент будет содержать представление контрагента с указанным значением ссылки.

Type

Данный суффикс используется для описания реквизита составного типа. Так, если в данных есть поле составного типа Контрагент, то в документе, который возвращает стандартный интерфейс OData, этому полю будет соответствовать два свойства:

- Контрагент_Type – будет содержать описание типа значения реквизита в виде строки (тип Edm.String, диспетчеризационное свойство);
- Контрагент – будет содержать значение реквизита (соответствующего типа).

Перечень допустимых типов, которые могут быть использованы в поле с таким суффиксом, определяется схемой сервиса, который можно получить при запросе полного описания стандартного интерфейса OData (см. здесь). Таким образом, при необходимости установить тип Документ.РасходТовара, в элемент с суффиксом _Type должно быть записано значение StandardODATA.Document_РасходТовара.

Если значение реквизита составного типа в информационной базе «1С:Предприятия» имеет значение Неопределено, то диспетчеризационное свойство будет иметь значение StandardODATA.Undefined, а само значение свойства должно игнорироваться.

Пример представления реквизита составного типа:

```
<d:РеквизитСоставной/>
<d:РеквизитСоставной_Type>StandardODATA.Undefined</d:РеквизитСоставной_Type>
```

Base64Data

Данный суффикс используется при указании имени свойства, содержащего данные, расположенные в реквизите типа ХранилищеЗначения, в виде строки Base64. Так, если в объекте конфигурации есть реквизит Файл, который имеет тип ХранилищеЗначения, то в документе, который возвращает стандартный интерфейс OData, этому полю будет соответствовать два свойства:

- Файл_Type – содержит наименование типа данных, хранимых реквизитом;
- Файл_Base64Data – содержит строку Base64, содержащую сами данные.

_____Presentation

Свойство <Имя свойства>_____Presentation содержит представление реквизита (между именем свойства и Presentation находятся 4 символа «_»). Значение поля эквивалентно значению, которое является результатом работы функции языка запросов `Представление()`.

Поля представлений не включаются в результаты запроса с пустым параметром $select и при указании в параметре $select значений * и **. Таким образом, если необходимо получить все реквизиты объекта и представления некоторых реквизитов, необходимо указать значение *, а затем отдельно перечислить поля представлений требуемых реквизитов, разделяя их запятыми: $select=*, РеквизитСсылка_____Presentation.

Для получения представлений всех реквизитов объекта, следует в параметре $select указать значение *_____Presentation: $select=*_____Presentation.

Для получения значений всех полей и представлений для всех полей, следует использовать следующее выражение: $select=*, *_____Presentation.

Для получения представления текущей сущности имеется возможность указывать в списке запрашиваемых полей поле Presentation: $select=Ref_Key, Presentation. В этом случае будет получено значение ссылки и представление этой ссылки.

Поля представлений не содержатся в описании метаданных, которое возвращается в описании стандартного интерфейса OData (параметр $metadata). Получение представления не поддерживается для табличных частей и реквизитов типа `ХранилищеЗначения`. Не поддерживается модификация значений для полей представлений.

17.2.1.4. Правила формирования имени ресурса

При обращении к какому-либо ресурсу, его идентификатор формируется по следующему принципу: ПрефиксИмени_ИмяОбъектаКонфигурации_СуффиксИмени. С помощью стандартного интерфейса OData можно получить доступ к следующим объектам (ПрефиксИмени):

Объект конфигурации	Префикс имени для указания в URL
Справочник	Catalog
Документ	Document
Журнал документов	DocumentJournal
Константа	Constant
План обмена	ExchangePlan
План счетов	ChartOfAccounts
План видов расчета	ChartOfCalculationTypes
План видов характеристик	ChartOfCharacteristicTypes
Регистр сведений	InformationRegister
Регистр накопления	AccumulationRegister
Регистр расчета	CalculationRegister
Регистр бухгалтерии	AccountingRegister
Бизнес-процесс	BusinessProcess
Задача	Task

ИмяОбъектаКонфигурации – свойство Имя объекта конфигурации, как оно задано при разработке прикладного решения в конфигураторе.

СуффиксИмени – предназначено для уточнения имени ресурса и является необязательной частью имени. В качестве суффикса имени могут выступать следующие выражения:

- Имя табличной части объекта;
- Имя реквизита табличной части или набора записей;
- Имя виртуальной таблицы регистра;
- RowType;
- RecordType.

Далее будут более подробно рассмотрены вышеописанные уточнения имени ресурса:

Имя табличной части объекта

Если объект обладает табличной частью, то для получения доступа ко всем записям этой табличной части необходимо добавить имя табличной части после имени самого объекта, например, для получения всех строк табличных частей `Товары` всех документов `РасходТовара` будет необходимо выполнить GET-запрос по следующему адресу: http://host/base/odata/standard.odata/Document_РасходТовара_Товары.

Имя реквизита табличной части или набора записей

Если объект обладает табличной частью, то имеется возможность указать, что требуется получение не всех реквизитов табличной части, а некоторого списка этих реквизитов. Для этого необходимо указать в параметре $select список требуемых реквизитов в следующем виде: <Имя табличной части>/<Имя поля>. Аналогичная возможность предоставляется для наборов записей регистров, где в качестве имени табличной части выступает RecordSet: RecordSet/<Имя поля>.

Имя виртуальной таблицы регистра

В роли виртуальной таблицы регистра выступает функция, связанная с ресурсом, возвращающей набор сущностей регистра. Имя функции совпадает с английским вариантом имени используемой виртуальной таблицы языка запросов. Параметры функции

соответствуют параметрам виртуальной таблицы. Так, для получения среза последних регистра сведений КурсыВалют, следует выполнить GET-запрос по следующему адресу: http://localhost/demo/odata/standard.odata/InformationRegister_КурсыВалют/SliceLast().

RowType

Сущность с таким суффиксом описывает тип строки табличной части какого-либо объекта.

RecordType

Сущность с таким суффиксом описывает отдельную запись регистра.

17.2.1.5. Правила формирования условия отбора

17.2.1.5.1. Общая информция

В данном разделе приводится информация по различным способам формирования отбора получаемых данных, используемых в стандартном интерфейсе OData системы «1С:Предприятие».

Описание возможностей отбора в протоколе OData можно получить в документации по протоколу (на английском языке):

- http://www.odata.org/documentation/odata-v3-documentation/odata-core/#10231_The_filter_System_Query_Option;
- http://www.odata.org/documentation/odata-v3-documentation/url-conventions/#512_Filter_System_Query_Option.

17.2.1.5.2. $filter

При получении данных можно фильтровать данные. Для этого предназначен специальный язык, который позволяет описывать условия, каким должны соответствовать данные, которые возвращает стандартный интерфейс OData. Описание отбора начинается с ключевого слова $filter, после которого следует собственно условие. Поддерживаются следующие операции:

- Логические операции:

Описание	Имя	Пример
Равно	eq	/Catalog_Города?$filter=Description eq 'Главный'
Не равно	ne	/Catalog_Города?$filter=Description ne 'Пермь'
Больше	gt	/Catalog_Товары?$filter=Цена gt 10
Больше или равно	ge	/Catalog_Товары?$filter=Цена ge 10
Меньше	lt	/Catalog_Товары?$filter=Цена lt 10
Меньше или равно	le	/Catalog_Товары?$filter=Цена le 10
Логическое ИЛИ	or	/Catalog_Товары?$filter=Цена lt 10 or Цена gt 100
Логическое И	and	/Catalog_Товары?$filter=Цена gt 10 and Цена lt 100
Отрицание	not	/Catalog_Товары?$filter=not (Цена eq 10)

- Арифметические операции:

Описание	Имя	Пример
Сложение	add	/Catalog_Товары?$filter=Цена add 5 gt 10
Вычитание	sub	/Catalog_Товары?$filter=Цена sub 5 gt 10
Умножение	mul	/Catalog_Товары?$filter=Цена mul 5 gt 1000
Деление	div	/Catalog_Товары?$filter=Цена div 4 gt 2

- Группирующие операторы:

Описание	Имя	Пример
Приоритет операции	()	/Catalog_Товары?$filter=(Цена add 5) gt 10

Пример отбора:

```
http://host/odata/standard.odata/Catalog_Товары?$filter=Имяeq'Молоко'andЦенаlt2500
```

При формировании условия отбора следует учитывать приоритет операций. В следующей таблице представлен список операций языка выражений отбора в порядке уменьшения приоритета. Операции с одинаковым приоритетом вычисляются слева направо:

Оператор	Описание
( )	Повышение приоритета операции
/	Навигация
-	Арифметическое отрицание
Not	Логическое отрицание
Mul	Умножение
Div	Деление

Add	Сложение
Sub	Вычитание
Gt	Больше
Ge	Больше или равно
Lt	Меньше
Le	Меньше или равно
Eq	Равно
Ne	Не равно
And	Логическое «И»
Or	Логическое «ИЛИ»

При формировании условий запроса (параметр filter) или формировании реквизита, по которому выполняется упорядочивание (параметр orderby) могут применяться следующие функции:

- Строковые функции:

Функция	Описание	Пример
substringof(Str1, Str2)	Возвращает true в том случае, если Str1 является подстрокой Str2.	/Catalog_Товары?$filter=substringof('Красный Октябрь', Производитель) eq true
endswith(Str1, Str2)	Возвращает true в том случае, если Str1 заканчивается на Str2.	/Catalog_Товары?$filter=endswith(Производитель, 'ООО') eq true
startswith(Str1, Str2)	Возвращает true в том случае, если Str1 начинается на Str2.	/Catalog_Товары?$filter=startswith(Производитель, 'ООО') eq true
substring(Str, Int1) substring(Str, Int1, Int2)	Возвращает подстроку из Str1. В варианте с двумя параметрами возвращается строка с позиции Int и до конца строки. В варианте с тремя параметрами возвращается подстрока, начиная с позиции Int1 и длиной Int2.	/Catalog_Поставщики?$filter=substring(ИНН, 1, 2) eq '77'
concat(Str1, Str2)	Возвращает строку, являющуюся результатом конкатенации Str1 и Str2.	/Catalog_Поставщик?$filter=concat(concat(Город, ', '), Страна) eq 'Москва, Россия'
like(Str, Template)	Возвращает true, если значение Str1 удовлетворяет шаблону Template. Синтаксис шаблона аналогичен функции ПОДОБНО() языка запросов (см. здесь).	/Catalog_Товары?$filter=like(Наименование, '[^к]%')

- Функции работы с датами:

Функция	Описание	Пример
year(DateTime)	Возвращает год из значения типа Edm.DateTime или Edm.DateTimeOffset.	/Catalog_Товары?$filter=year(Произведен) eq 2013
quarter(DateTime)	Номер квартала года, в котором находится указанное значение типа Edm.DateTime.	/Catalog_Товары?$filter=quarter(ДатаПроизводства) eq 1
month(DateTime)	Возвращает месяц из значения типа Edm.DateTime или Edm.DateTimeOffset.	/Catalog_Товары?$filter=month(Произведен) eq 12
day(DateTime)	Возвращает день из значения типа Edm.DateTime или Edm.DateTimeOffset.	/Catalog_Товары?$filter=day(Произведен) eq 1
hour(DateTime)	Возвращает значение часов из значения типа Edm.DateTime или Edm.DateTimeOffset.	/Catalog_Товары?$filter=hour(Произведен) eq 23
minute(DateTime)	Возвращает значение минут из значения типа Edm.DateTime или Edm.DateTimeOffset.	/Catalog_Товары?$filter=minute(Произведен) eq 59
second(DateTime)	Возвращает значение секунд из значения типа Edm.DateTime или Edm.DateTimeOffset.	/Catalog_Товары?$filter=second(Произведен) eq 59
datedifference(DateTime1, DateTime2, Type)	Возвращает разность дат DateTime2 и DateTime1 в единицах, указанных параметром Type: • second – секунды; • minute – минуты; • hour – часы; • day – дни; • month – месяцы; • quarter – кварталы;	/Catalog_Товары?$filter=datedifference(Произведен, ГоденДо, 'day') gt 10

	• year – года.	
dateadd(DateTime1, Type, Int1)	Возвращает дату, полученную добавлением к значению DateTime1 значения Int1, выраженное в единицах Type: • second – секунды; • minute – минуты; • hour – часы; • day – дни; • month – месяцы; • quarter – кварталы; • year – года.	/Catalog_Товары?$filter=dateadd(Произведен, 'month', 1) eq ГоденДо
dayofweek(DateTime)	Возвращает день недели по значению типа Edm.DateTime.	/Catalog_Товары?$filter=dayofweek(ДатаПроизводства) eq 7
dayofyear(DateTime)	Возвращает день года по значению Edm.DateTime.	/Catalog_Товары?$filter=dayofyear(ДатаПроизводства) eq 1

- Прочие функции:

Функция	Описание	Пример
round(Number)	Возвращает параметр, округленный до ближайшего целого числа.	/Catalog_Товары?$filter=round(Вес) gt 31
isof(expr, type)	Возвращает true в том случае, когда объект, на который указывает параметр expr, имеет тип, на который ссылается type. В качестве типа значения принимается строка, обозначающая имя данного типа, например: String, Number, Boolean, Date, Catalog_Товары и т. д.	/Catalog_Товары?$filter=isof(Цена, 'Number')
cast(expr, type)	Возвращает объект, на который указывает параметр expr, приведенный к типу, указанному параметром type. В качестве типа значения принимается строка, обозначающая имя данного типа, например: String, Number, Boolean, Date, Catalog_Товары и т. д.	/Catalog_Товары?$filter=cast(РеквизитСоставной, 'Number') le 12

ПРИМЕЧАНИЕ. Не поддерживаются операции сравнения с реквизитом типа ХранилищеЗначения.

Имеется возможность выполнять отбор сущностей при помощи проверки на равенство поля составного типа и ссылки. Для этого следует использовать функцию cast().

Пример:

```
$filter=ДокументПриходаeqcast(guid'0d4a79cb-9843-4147-bcd9-80ac3ca2b9c7','Document_ПриходнаяНакладная')
```

В данном примере у используемой сущности имеет реквизит составного типа ДокументПрихода. Запрос будет отбирать все записи сущности, у которой данный реквизит заполнен ссылкой на документ ПриходнаяНакладная с указанным уникальным идентификатором (0d4a79cb-9843-4147-bcd9-80ac3ca2b9c7).

Если требуется выполнить отбор по реквизиту типа УникальныйИдентификатор, то это выполняется с помощью простой операции сравнения:

```
$filter=ИмяРеквизитаeqguid'0d4a79cb-9843-4147-bcd9-80ac3ca2b9c7'
```

Для отбора по типу какого-либо реквизита составного типа следует использовать функцию isof():

```
$filter=isof(ДокументыПрихода,'Document_ПриходнаяНакладная')
```

Следующее выражение всегда вернет пустой результат запроса, вне зависимости от физического наличия используемого уникального идентифкатора в соответствующей колонке сущности:

```
$filter=ДокументыПриходаeqguid'0d4a79cb-9843-4147-bcd9-80ac3ca2b9c7'
```

При необходимости выполнить отбор по элементу коллекции, необходимо использовать лямбда-функции. В качестве аргументов лямбда-функции выступает имя лямбда-переменной, за которой следует символ двоеточия и логическое выражение, которое использует лямбда-переменную для указания на свойства элемента коллекции. Система поддерживает использование следующих лямбда-функций:

- any – применяет логическое выражение к каждому элементу коллекции и возвращает значение true, если хоть один элемент коллекции удовлетворяет этому условию. Лямбда-функция any без аргументов возвращает true, если коллекция не пуста.

```
http://host/base/odata/standard.odata/Document_Продажи?$filter=Товары/any(d:d/Ценаgt10000)
```

В приведенном примере формируется список документов Продажи, у которых есть табличная часть Товары, в которой есть реквизит Цена. При этом Цена должна быть больше, чем 10000. В результирующий список попадут документы, в состав которых входит хотя бы одна строка, удовлетворяющая условию.

- all – применяет логическое выражение к каждому элементу коллекции и возвращает значение true, если все элементы коллекции ему удовлетворяет.

```
http://host/base/odata/standard.odata/Document_Продажи?$filter=Товары/any(d:d/Ценаlt10000)
```

В приведенном примере формируется список документов Продажи, у которых есть табличная часть Товары, в которой есть реквизит Цена. При этом Цена должна быть меньше, чем 10000. В результирующий список попадут документы, в составе которых все строки удовлетворяют заданному условию.

При необходимости указать условие на значение реквизита другого реквизита составного типа, содержащего ссылочные значения, таким образом, чтобы проверялось одновременно и значение проверяемого реквизита и тип составного реквизита, требуется использовать функцию cast().

Например, имеется документ МаршрутныйЛист, который содержит реквизит составного типа ОснованиеОтгрузки. Данный реквизит может принимать значения типа ДокументСсылка.Накладная и ДокументСсылка.ВнутреннееПеремещение. При этом у документа Накладная существует реквизит МестоОтгрузки, который может принимать значения типа СправочникСсылка.Склад и СправочникСсылка.АдресаОтгрузки.

В том случае, если необходимо отобрать документы МаршрутныйЛист с условием по названию склада, указанного в накладной (т.е. по реквизиту Название объекта типа СправочникСсылка.Склад, записанного в реквизит МестоОтгрузки документа Накладная), то условие должно выглядеть следующим образом:

```
cast(cast(ОснованиеОтгрузки,'Document_Накладная')/МестоОтгрузки,'Catalog_Склад')/Названиеeq'ОсновнойвМоскве'
```

Не поддерживаются следующие стандартные функции: lenght, indexof, replace, tolower, toupper, trim, years, days, hours, seconds, floor, ceiling;

17.2.1.5.3. $top

Имеется возможность ограничить количество записей, возвращаемых при обращении к ресурсу. Для этого используется параметр $top.

Пример:

```
http://host/odata/standard.odata/Catalog_Товары?$filter=Ценаlt1000&$top=10
```

17.2.1.5.4. allowedOnly

Если при выполнении запроса необходимо получить только те объекты данных, которые не попадают под ограничения доступа к данным, то в URL получения данных необходимо добавить параметр allowedOnly.

Пример:

```
http://host/base/odata/standard.odata/Catalog_Товары?$allowedOnly=true
```

Если параметр не указан или указан со значением false, то во время исполнения запроса к данным может возникнуть ошибка с кодом 401 (если результат выполнения запроса содержит данные, доступ к которым запрещен). Ошибка может не произойти в том случае, если были указаны дополнительные условия, которые ограничили выборку только разрешенными данными.

Например, для справочника Данные настроено ограничение доступа к данным, которое не позволяет получить элементы справочника, у которых реквизит ЗначениеСекретности равно 1, но позволяет получить элементы справочника, у которых реквизит ЗначениеСекретности равно 2. В этом случае следующий запрос не приведет к возникновению ошибки, т. к. явно накладывается условие, которое оставляет в выборе только разрешенные данные:

```
http://host/base/odata/standard.odata/Catalog_Данные?allowedOnly=false&$filter=ЗначениеСекретностиeq2
```

Параметр allowedOnly можно использовать только для GET-запросов к наборам сущностей.

17.2.1.5.5. $skip

Позволяет исключить из результата запроса первые несколько записей. Если параметры $top и $skip указываются одновременно, то параметр $skip будет применен раньше, чем параметр $top. Приоритет применения параметров не зависит от порядка их указания в теле запроса.

Пример:

```
http://host/base/odata/standard.odata/Catalog_Товары?$skip=2
```

17.2.1.6. Параметры запроса

17.2.1.6.1. $count

Данный параметр позволяет получить в качестве результата запроса не выборку, а ее (выборки) размер. При успешном выполнении, тело ответа должно содержать только число элементов коллекции, отформатированное как обычное число.

Пример:

```
//возвращаетколичествоэлементовсправочникаТовары,
//длякоторыхсправедливоусловие:значениереквизитаЦенабольше500
http://host/base/odata/standard.odata/Catalog_Товары/$count?$filter=Ценаgt500
```

17.2.1.6.2. $inlinecount

Параметр позволяет указать, что система должна включить результат запроса не только полученные записи, но и количество этих записей. Для этого необходимо указывать значение параметра $inlinecount, равное allpages. Использование параметра $inlinecount со значением none подавляет возвращение количества записей вместе с результатом запроса. Если параметр $inlinecount не указан в теле запроса, то количество записей вместе с результатом запроса не возвращается. Использование параметра $inlinecount приводит к игнорированию параметров $skip и $top. Допускается совместное использование параметров $filter и $inlinecount.

Пример:

```
http://host/base/odata/standard.odata/Catalog_Товары?$inlinecount=allpages
```

Пример результата запроса, с параметром $inlinecount=allpages:

Формат atom-xml:

```
<?xmlversion="1.0"encoding="UTF-8"?>
<feedxmlns=http://www.w3.org/2005/Atomxmlns:at=http://purl.org/atompub/tombstones/1.0-
xmlns:d=http://schemas.microsoft.com/ado/2007/08/dataservices-
xmlns:m=http://schemas.microsoft.com/ado/2007/08/dataservices/metadata-
xml:base="http://localhost/ODataTests/odata/standard.odata/">
^<entry>
^^…
^</entry>
^^…
^<entry>
^^…
^</entry>
^<m:count>19</m:count>
</feed>
```

Формат json:

```
{
^"odata.metadata":"http://localhost/ODataTests/odata/standard.odata/$metadata#Document_ДемоДокумент",
^"odata.count":"19",
^"value":[{...},...,{...}]
}
```

17.2.1.6.3. $orderby

Данный параметр позволяет задать упорядочивание результата запроса. Значение параметра $orderby содержит список реквизитов, разделенных запятыми. Значение реквизита может включать направление упорядочивания:

- asc – для упорядочивания по возрастанию;
- desc – для упорядочивания по убыванию.

Направление упорядочивания должно быть отделено от имени реквизита на 1 и более пробелов. Если asc или desc не указан, то считается, что требуется упорядочивание по возрастанию.

Пример:

```
http://host/base/odata/standard.odata/Catalog_Товары?$orderby=Названиеasc,Производительdesc
```

Допускается упорядочивание по свойствам дочерних реквизитов. В этом случае полное имя реквизита, по которому выполняется упорядочивание, формируется с использованием разделителя "/".

Пример:

```
http://host/base/odata/standard.odata/Document_Расход?$orderby=Контрагент/ИННasc
```

В этом примере упорядочивание будет произведено по возрастанию (asc) значения реквизита ИНН у сущности, на которую ссылается реквизит Контрагент документа Расход.

17.2.1.6.4. $expand

Данный параметр позволяет вместе с результатами основного запроса получать значения связанных сущностей, что позволит не запрашивать каждую сущность отдельно.

Значением параметра выступает список из навигационных ссылок или составных реквизитов (если составные реквизиты могут принимать ссылочное значение), разделенных запятыми. Для расширения реквизитов составных реквизитов, после имени составного реквизита должно идти название типа, на который может хранить ссылку составной реквизит, а уже после этого сам расширяемый реквизит. Тогда, если у сущности составной реквизит имеет ссылку на объект указанного типа, то в нем будет расширен указанный реквизит. Если же значение является ссылкой на объект иного типа или не является ссылкой, то расширение проводиться не будет.

Возможно использование опции для расширения сущностей из результатов функций SliceLast() и SliceFirst().

Пример:

```
http://host/base/odata/standard.odata/Catalog_Клиенты?$expand=Сертификат
```

Для каждого элемента из справочника Клиенты, в результат запроса должен быть помещен объект, на который ссылается реквизит Сертификат.

Пример:

```
http://host/base/odata/standard.odata/Catalog_Клиенты?$expand=Договор/Document_ДоговорНайма/Заявление
```

Для каждого элемента из справочника Клиенты, в результат запроса должен быть добавлен объект, на который ссылается реквизит составного типа Договор (если он содержит какое-либо ссылочное значение). Если реквизит Договор хранит ссылку на объект типа ДокументСсылка.ДоговорНайма, то объект, на который ссылается Договор.Заявление так же должен быть добавлен в результате запроса.

Значение * в конце пути параметра $expand означает необходимость раскрыть все навигационные ссылки внутри сущности. Однако, нельзя раскрывать навигационные поля у составного реквизита без явного указания типа ссылки.

Пример:

```
http://host/base/odata/standard.odata/Catalog_Клиенты?$expand=Договор/Document_ДоговорНайма/*
```

В данном примере Договор – это составной реквизит. Данный вариант указания * поддерживается.

Пример:

```
http://host/base/odata/standard.odata/Catalog_Клиенты?$expand=Договор/*
```

В данном примере Договор – это составной реквизит. Данный вариант указания * не поддерживается.

Примеры:

```
http://host/base/odata/standard.odata/Catalog_Клиенты?$expand=*
```

Для каждого элемента из справочника Клиенты, в результат запроса должны быть добавлены объекты, на которые ссылаются все навигационные ссылки объекта Клиента.

Пример:

```
http://host/base/odata/standard.odata/Catalog_Клиенты?$expand=Договор/*
```

Для каждого элемента из справочника Клиенты, в результат запроса должны быть добавлены объекты, на которые ссылаются все навигационные ссылки внутри объекта Договор и сам объект Договор. При этом реквизит Договор является реквизитом ссылочного типа.

Ограничения параметра $expand:

- Не поддерживается расширение реквизитов табличных частей.
- Не поддерживается расширение при запросе одиночных сущностей (сущность, строки табличной части, запись регистра).
- Расширение ссылочных и составных типов виртуальных таблиц не соответствует протоколу OData версии 3.

Запрос, в котором будет выполняться расширение измерения виртуальной таблицы будет выглядеть следующим образом:

```
http://host/base/odata/standard.odata/AccountingRegister_РегистрБухгалтерии/Balance?$expand=Организация
```

В результате исполнения этого запроса будет получен следующий результат:

```
<d:Организация_Key>УникальныйИдентификатор</d:Организация_Key>
<d:Организация>
^…ДанныеОбъекта…
</d:Организация>
```

При расширении составного типа поведение аналогично, за исключением того, что данные объекта будут представлены в элементе с суффиксом _Expanded.

17.2.1.7. Способы получения описания стандартного интерфейса OData

Для получения упрощенного описания стандартного интерфейса OData (только список сущностей) необходимо выполнить GET-запрос с использованием URL http://host/base/odata/standard.odata.

Для получения описания стандартного интерфейса OData необходимо выполнить GET-запрос с использованием URL http://host/base/odata/standard.odata/$metadata. В результате будет получен полный список доступных сущностей, их атрибутов и функций в виде XML-документа. Подробное описание документа можно получить по адресу http://www.odata.org/documentation/odata-v3-documentation/common-schema-definition-language-csdl/ (на английском языке).

В случае получение данных в формате **json**, не поддерживается запрос получения расширенного описания метаданных вида http://host/base/odata/standard.odata/$metadata. Однако имеется возможность управлять детализацией информации о метаданных, которая возвращается при получении данных (как сущностей, так и списков сущностей). Для этого необходимо особым образом формировать параметр $format при выполнении запроса:

- $format=application/json;odata=minimalmetadata – в этом случае информация о метаданных передается в минимальном объеме. Это значение по умолчанию.
- $format=application/json;odata=nometadata – в этом случае информация о метаданных не передается вовсе.
- $format=application/json;odata=fullmetadata – в этом случае информация о метаданных передается в полном объеме.

17.2.1.8. Способы получения данных

С помощью стандартного интерфейса OData можно получать как списки сущностей, так и сами сущности. Эти способы получения данных отличаются URL, по которому происходит обращение к данным.

Для получения списка сущностей URL выглядит следующим образом: http://host/base/odata/standard.odata/Catalog_Товары. Это **URL набора сущностей**. Для получения сущности URL будет выглядеть следующим образом: http://host/base/odata/standard.odata/Catalog_Товары(guid'value'). В общем виде такой адрес будет называться **канонический URL экземпляра сущности**. Разница заключается в том, что в первом случае получается вся таблица справочника товаров, а вот втором выбирается один объект, который описывается набором ключевых значений (в примере ключевое значение единственное – ссылка на объект). В конкретном случае набор ключевых полей, описывающих сущность, зависит от получаемой сущности. Если ключевых полей более одного, то они должны быть указаны все. Ключевые значения указываются парами Имя=Значение, разделенными запятыми. Если ключевое значение одно – имя ключа можно не указывать. Например, для получения значения курса валюты на конкретную дату необходимо выполнить GET-запрос со следующим URL:

```
http://host/base/odata/standard.odata/InformationRegister_КурсыВалют(Period=datetime'2008-02-05T00:00:00',-
Валюта_Key=guid'9d5c4222-8c4c-11db-a9b0-00055d49b45e')
```

При получении данных существует возможность указать, какие свойства сущности (или набора сущностей) необходимо получить с помощью стандартного интерфейса OData. Для этого существует ключевое слово $select, которое имеет несколько вариантов использования:

- Признак получения всех свойств.

  В этом случае следует указать выражение вида $select=* или совсем не указывать выражение $select.

- Перечень конкретных свойств, которые необходимо получить.

В этом случае необходимо указать в выражении оператора $select перечень (через запятую) всех необходимых полей. При этом следует указывать суффиксы свойств, если это необходимо. Например, для получения значения ссылки на справочник и свойств Код и Артикул, для справочника Товары, необходимо использовать следующий URL:

```
http://host/base/odata/standard.odata/Catalog_Товары?$select=Ref_Key,Код,Артикул
```

Для получения этих же полей для конкретного экземпляра сущности (с известным идентификатором) будет использоваться следующий URL:

```
http://host/base/odata/standard.odata/Catalog_Товары(guid'value')?$select=Ref_Key,Код,Артикул
```

- Признак получения всех свойств, кроме табличных частей.

Для этого в выражении оператора $select необходимо указать значение **. При использовании следующего URL будут получены все документы РасходТовара со всеми реквизитами, кроме табличных частей:

```
http://host/base/odata/standard.odata/Document_РасходТовара?$select=**
```

- Указание списка для получения связанных сущностей.

Допускается совместное использование параметров запроса $select и $expand. В этом случае с помощью параметра $expand указываются те свойства получаемой сущности, которые следует поместить в результат запроса. Конкретные значения получаемых (или разворачиваемых) свойств указывается с помощью параметра $select. Описание работа с параметром $expand более подробно см. здесь. В параметре $select, при использовании совместно с параметром $expand, поддерживается использование символов * и **. Указание символа * означает, что необходимо развернуть все реквизиты получаемого объекта. Указание символа ** означает, что необходимо развернуть все реквизиты получаемого объекта, кроме табличных частей.

```
http://host/base/odata/standard.odata/Catalog_Клиенты?$expand=Сертификат&$select=Сертификат/Дата
http://host/base/odata/standard.odata/Catalog_Клиенты?$expand=Договор/Document_ДоговорНайма/
Заявление&$select=Договор/Document_ДоговорНайма/Заявление/**
http://host/base/odata/standard.odata/InformationRegister_КурсыВалют?$expand=Валюта&$select=Период,Валюта/Код,Курс
http://host/base/odata/standard.odata/Catalog_Клиенты?$expand=Сертификат&$select=Сертификат/*
```

Если в параметре $select одновременно указаны поля вида Объект/*, Объект/**, Объект/ИмяРеквизита, то они обрабатываются в следующем порядке:

1. Если указан символ *, то включаются все реквизиты;

2. Иначе, если указан символ **, то включаются все реквизиты, кроме табличных частей, иные указания игнорируются. Таким образом, если указано выражение вида $select=Объект/**, Объект/ТабличнаяЧасть, то табличная часть включена не будет.

3. Иначе, включаются все указанные реквизиты.

Имеется возможность получать данные, доступные «через точку». В этом случае к адресу, описывающему **конкретную** сущность, прибавляются имена свойств, идущие «через точку», но разделенные символом «/». При этом для стандартных реквизитов следует использовать только английские имена реквизитов. Например, для получения свойства Наименование реквизита Валюта документа РасходТовара, необходимо использовать GET-запрос со следующим URL:
http://host/base/odata/standard.odata/Document_РасходТовара(guid'value')/Валюта/Description.

17.2.1.9. Выполнение функций и действий

С некоторыми сущностями и наборами сущностей могут быть связаны функции. Например, через функции выполняется работа с виртуальными таблицами регистров. В этом случае URL ресурса формируется следующим образом:
http://host/base/odata/standard.odata/<ресурс>/<функция>(<параметры>). Подробнее рассмотрим, из чего состоит адрес. Первая часть адреса (http://host/base/odata/standard.odata) представляет собой стандартный префикс адреса при обращении к стандартному интерфейсу OData. <ресурс> – имя ресурса, правила формирования которого см. здесь. Имя функции соответствует англоязычному имени виртуальной таблицы языка запросов (см. здесь). <параметры> у функции задаются парами Ключ=Значение и разделяются запятыми. Если в качестве параметра функции используется отбор, то выражение, описывающее отбор, должно удовлетворять общим правилам описания отборов (см. здесь). Так, получение среза первых для регистра курсов валют (с параметрами), будет выполняться по следующему URL: http:// host/base/odata/standard.odata/InformationRegister_КурсыВалют/SliceFirst(Period=datetime'2008-01-01T00:00:00',Condition='Валюта_Key eq guid'value'').

17.2.1.10. Ошибочные ситуации

В случае ошибочной ситуации возвращается ответ с HTTP-статусом 4XX или 5XX. Статус 4XX информирует об ошибках на стороне клиентского приложения, статус 5XX информирует об ошибке на стороне сервера.

В случае статуса 4XX сервер пытается уточнить причину ошибки и может передать клиентскому приложению дополнительный внутренний код ошибки и информационное сообщение (в виде xml-документа) в теле ответа.

Пример:

```
<m:errorxmlns:m="http://schemas.microsoft.com/ado/2007/08/dataservices/metadata">
^<m:code>9</m:code>
^<m:message>Экземплярсущности"НакладнаяОтгрузки"ненайденпопереданномуключу.</m:message>
</m:error>
```

Далее перечислены внутренние коды ошибок с описанием причины появления:

Код	Описание
0	Возможность не поддерживается
1	Не удалось разобрать строку
2	Неверный формат запроса
3	Запрошенный тип представления не поддерживается
4	Неверное значение свойства
5	Отсутствует обязательное значение свойства

6	Неверный URL
7	Не хватает элемента ключа сущности
8	Тип сущности не найден
9	Экземпляр сущности не найден
10	Запрошенное свойство не найдено
11	Метод не найден
12	Отсутствует обязательный аргумент метода
13	Создание строк табличных частей напрямую не поддерживается
14	Ошибка разбора опций запроса
15	Сущность с таким ключом уже существует
16	Не удалось присвоить свойство
17	Объект не поддерживает режим загрузки данных
18	Ошибка инициализации интерфейса OData: в объекте есть свойства с одинаковыми именами
19	Использованный HTTP-метод запрещен в данном контексте
20	Ошибка прав доступа. Может возникать: • Когда у пользователя нет прав на запрошенное действие над данным объектом; • Когда в выборку попадает объект, недоступный в связи с ограничением доступа к данным и параметр allowedOnly не используется.
21	Вызов нереализованной функции. Указание неверного количества аргументов функции. Попытка передачи аргумента неверного типа. Указание нереализованной лямбда-функции.

17.2.1.11. Установка значений разделителей

Значения разделителей можно устанавливать только следующими способами:

- С помощью фрагментов URL при обращении к стандартному интерфейсу OData;
- С помощью файла описания публикации default.vrd.

Указанием разделителей с помощью параметры командной строки Z не поддерживается.

17.2.1.12. Публикация стандартного интерфейса OData

Публикация стандартного интерфейса OData выполняется с помощью диалога публикации на веб-сервере (Администрирование – Публикация на веб-сервере) и описано в книге 1С:Предприятие 8.3. "Руководство администратора".

Для того чтобы объекты конфигурации стали доступны через стандартный интерфейс OData, необходимо разрешить это с помощью метода глобального контекста `УстановитьСоставСтандартногоИнтерфейсаOData()`. Если прикладное решение функционирует в режиме совместимости с версией 8.3.4 (и ниже), данный метод не используется. В этом случае с помощью стандартного интерфейса OData доступны все поддерживаемые объекты конфигурации.

С помощью данного метода можно ограничить перечень доступных объектов конфигурации только теми (из общего перечня поддерживаемых, см. здесь), доступ к которым действительно необходим для внешних приложений.

Механизм установки состава объектов, доступных с помощью стандартного интерфейса OData, можно выполнить в виде внешней обработки. Для этого не требуется модифицировать прикладное решение.

17.2.1.13. Выполнение типовых операций

В данном разделе будут приведены примеры выполнения некоторых типовых операций по работе с данными с помощью стандартного интерфейса OData.

> **ВНИМАНИЕ!** Данные примеры не являются законченными. Они приведены только для иллюстрации применения различных конструкций.

17.2.1.13.1. Работа с одним объектом

Чтение данных

Для получения информации о сущности следует использовать канонический URL сущности. Чтение выполняется с помощью GET-запроса.

Пример чтения ссылочного объекта:

```
http://host/base/odata/standard.odata/Catalog_Товары(guid'value')
```

Пример чтения набора записей подчиненного регистра сведений:

```
http://host/base/odata/standard.odata/InformationRegister_ПриходныеЦены(Recorder_Key=guid'value')
```

В данном примере guid'value' идентифицирует документ, выполнивший движение по регистру сведений.

Пример чтения записи регистра сведений:

```
http://host/base/odata/standard.odata/InformationRegister_ПриходныеЦены_RecordType(Товар_Key=guid'value',-
ТипЦены="Приходная")
```

Пример чтения строки конкретного документа:

```
http://host/base/odata/standard.odata/Document_РасходТовара_Товары(Ref_Key=guid'value',LineNumber=1)
```

Чтение виртуальной таблицы СрезПоследних регистра сведений КурсыВалют (с отбором по измерению ОсновнаяВалюта типа СправочникСсылка.Валюты) будет выглядеть следующим образом:

Пример чтения независимого регистра сведений:

```
http://host/base/odata/standard.odata/InformationRegister_КурсыВалют/SliceLast?Condition=Валюта/ОсновнаяВалюта_Keyeq-
guid'value'
```

Пример чтения подчиненного регистра сведений (к имени регистра следует добавить _RecordType):

```
http://host/base/odata/standard.odata/InformationRegister_КурсыВалют_RecordType/SliceLast?Condition=Валюта/
ОсновнаяВалюта_Keyeqguid'value'
```

В данном примере guid'value' идентифицирует элемент справочника Валюты, по значению которого выполняется отбор.

Пример получения остатков на счете из регистра бухгалтерии с условием по конкретному контрагенту:

```
http://host/base/odata/standard.odata/AccountingRegister_Хозрасчетный/Balance(AccountCondition='Account_Keyeq-
guid'value1'',Condition='ExtDimension1eqcast(guid'value2','Catalog_Контрагенты')',ExtraDimensions='value3')?
$format=json
```

В данном примере:

- guid'value1' – уникальный идентификатор элемента плана счетов;
- guid'value2' – уникальный идентификатор элемента плана видов характеристик ВидыСубконтоХозрасчетные с типом значения СправочникСсылка.Контрагенты;
- 'value3' – уникальный идентификатор элемента справочника Контрагенты.

Создание объекта

Для создания объекта следует воспользоваться POST-запросом с использованием URL набора сущностей, передав в теле запроса документ (в поддерживаемом формате), который содержит значения полей создаваемого объекта. Если передаваемый документ содержит свойства, отсутствующие у создаваемого объекта, то эти свойства игнорируются.

Далее приводится пример создания элемента справочника Товары и ответ стандартного интерфейса OData после успешного выполнения операции.

Пример POST-запроса (формат atom):

```
POSThttp://host/base/odata/standard.odata/Catalog_ТоварыHTTP/1.1
User-Agent:Fiddler
Host:host
Content-Length:981
<entry>
^<categoryterm="StandardODATA.Catalog_Товары"scheme="http://schemas.microsoft.com/ado/2007/08/dataservices/scheme"/>
^<titletype="text"/>
^<updated>2014-02-14T12:05:55</updated>
^<author/>
^<summary/>^^
^<contenttype="application/xml">
^^<m:propertiesxmlns:d="http://schemas.microsoft.com/ado/2007/08/dataservices"-
xmlns:m="http://schemas.microsoft.com/ado/2007/08/dataservices/metadata">
^^^<d:DeletionMark>false</d:DeletionMark>
^^^<d:Parent_Key>bbb079ae-8c51-11db-a9b0-00055d49b45e</d:Parent_Key>
^^^<d:IsFolder>false</d:IsFolder>
^^^<d:Code>000000800</d:Code>
^^^<d:Description>Шлепанцы</d:Description>
^^^<d:Артикул>SL56X</d:Артикул>
^^^<d:Поставщик_Key>086715b0-f348-11db-a9c5-00055d49b45e</d:Поставщик_Key>
^^^<d:Вид>Товар</d:Вид>
^^^<d:Штрихкод/>
^^^<d:Описание>&lt;html&gt;Шлепанцыпляжные&lt;/html&gt;</d:Описание>
^^</m:properties>
^</content>
</entry>
```

Пример ответа стандартного интерфейса OData:

```
HTTP/1.1201Created
Content-Length:2705
Content-Type:application/atom+xml;type=entry;charset=utf-8
Location:http://host/base/odata/standard.odata/Catalog_Товары(guid'41aa6331-954f-11e3-814b-005056c00008')
Server:Microsoft-IIS/7.5
DataServiceVersion:3.0
X-Powered-By:ASP.NET
Date:Fri,14Feb201408:18:36GMT
<?xmlversion="1.0"encoding="UTF-8"?>
<entryxmlns="http://www.w3.org/2005/Atom"
^xmlns:d=http://schemas.microsoft.com/ado/2007/08/dataservices-
```

```
xmlns:m="http://schemas.microsoft.com/ado/2007/08/dataservices/metadata">
^<id>http://host/base/odata/standard.odata/Catalog_Товары(guid'41aa6331-954f-11e3-814b-005056c00008')</id>
^<categoryterm="StandardODATA.Catalog_Товары"scheme="http://schemas.microsoft.com/ado/2007/08/dataservices/scheme"/>
^<titletype="text"/>
^<updated>2014-02-14T12:18:36</updated>
^<author/>
^<summary/>
^<linkrel="http://schemas.microsoft.com/ado/2007/08/dataservices/related/ФайлКартинки"
^^^href="Catalog_Товары(guid'41aa6331-954f-11e3-814b-005056c00008')/ФайлКартинки"
^^^type="application/atom+xml;type=entry;charset=utf-8"
^^^title="ФайлКартинки"/>
^<linkrel="http://schemas.microsoft.com/ado/2007/08/dataservices/related/Parent"
^^^href="Catalog_Товары(guid'41aa6331-954f-11e3-814b-005056c00008')/Parent"
^^^type="application/atom+xml;type=entry;charset=utf-8"
^^^title="Parent"/>
^<linkrel="http://schemas.microsoft.com/ado/2007/08/dataservices/related/Поставщик"
^^^href="Catalog_Товары(guid'41aa6331-954f-11e3-814b-005056c00008')/Поставщик"
^^^type="application/atom+xml;type=entry;charset=utf-8"
^^^title="Поставщик"/>
^<linkrel="edit"
^^^href="Catalog_Товары(guid'41aa6331-954f-11e3-814b-005056c00008')"
^^^title="edit-link"/>
^<contenttype="application/xml">
^^<m:propertiesxmlns:d="http://schemas.microsoft.com/ado/2007/08/dataservices"-
xmlns:m="http://schemas.microsoft.com/ado/2007/08/dataservices/metadata">
^^^<d:Ref_Key>41aa6331-954f-11e3-814b-005056c00008</d:Ref_Key>
^^^<d:DataVersionm:null="true"/>
^^^<d:Description>Шлепанцы</d:Description>
^^^<d:Code>000000800</d:Code>
^^^<d:Parent_Key>bbb079ae-8c51-11db-a9b0-00055d49b45e</d:Parent_Key>
^^^<d:IsFolder>false</d:IsFolder>
^^^<d:DeletionMark>false</d:DeletionMark>
^^^<d:Артикул>SL56X</d:Артикул>
^^^<d:Поставщик_Key>086715b0-f348-11db-a9c5-00055d49b45e</d:Поставщик_Key>
^^^<d:ФайлКартинки_Key>00000000-0000-0000-0000-000000000000</d:ФайлКартинки_Key>
^^^<d:Вид>Товар</d:Вид>
^^^<d:Штрихкод/>
^^^<d:Описание>&lt;html&gt;Шлепанцыпляжные&lt;/html&gt;</d:Описание>
^^^<d:ФайлКартинки_Key>00000000-0000-0000-0000-000000000000</d:ФайлКартинки_Key>
^^^<d:Parent_Key>bbb079ae-8c51-11db-a9b0-00055d49b45e</d:Parent_Key>
^^^<d:Поставщик_Key>086715b0-f348-11db-a9c5-00055d49b45e</d:Поставщик_Key>
^^</m:properties>
^</content>
</entry>
```

Пример POST-запроса (формат json):

```
POSThttp://host/base/odata/standard.odata/Catalog_ТоварыHTTP/1.1
Accept:application/json
Accept-Charset:UTF-8
User-Agent:Fiddler
Content-Type:application/json
Content-Length:2426
{
"DeletionMark":false,
"Parent_Key":"bbb079ae-8c51-11db-a9b0-00055d49b45e",
"IsFolder":false
"Code":"000000800",
"Description":"Шлепанцы",
"Артикул":"SL56X",
"Поставщик_Key":"086715b0-f348-11db-a9c5-00055d49b45e",
"Вид":"Товар",
"Штрихкод":null,
"Описание:"Шлепанцыпляжные"
}
```

Обновление объекта

Для обновления объекта необходимо выполнить PUT-/PATCH-запрос с использованием канонического URL сущности (аналогично запросу GET для получения сущности), передав в теле запроса XML-документ (в формате atom) или JSON-документ (в формате json), который содержит значения свойств сущности. Если передаваемый документ содержит свойства, отсутствующие у создаваемого объекта, то эти свойства игнорируются.

В случае PATCH-запроса пропущенные свойства сущности будут проигнорированы, т. е. будут изменены только те свойства, которые переданы в запросе на изменение. Для PUT-запроса необходимо указывать значения всех свойств обновляемой сущности.

При модификации сущности через PUT-запрос, при записи значений ссылочных реквизитов, следует использовать поле ИмяСсылочногоРеквизита@odata.bind куда следует поместить URI сущности (можно в укороченном варианте – только фрагмент пути после standard.odata).

Например:

```
PUThttp://host/base/odata/standard.odata/Document_Документ(guid'19961ec3-3c5f-11e7-8785-50465da19fe4')?$format=json
{
^"Date":"2015-01-01T12:01:07",
^"Организация@odata.bind":"Catalog_Организации(guid'298584b5-92e4-11e3-8bc3-0050568b1678')",
^"Склад@odata.bind":http://host/base/odata/standard.odata/Catalog_Склады(guid'081bc346-2abc-13e4-a1bd-0050568b1688')
^…
}
```

В следующем примере рассматривается изменение реквизита Наименование и состава табличной части ТорговыеЗалы справочника Магазины. Остальные реквизиты справочника не используются в рамках данного примера. В примере используется формат данных json.

Пример PATCH-запроса:

```
PATCHhttp://host/base/odata/standard.odata/Catalog_Магазины(guid'value')?$format=jsonHTTP/1.1
Host:host
Connection:keep-alive
Accept:application/json
Content-Length:638
{
^"odata.metadata":"http://host/base/odata/standard.odata/$metadata#Catalog_Магазины/@Element",
^"Description":"Новоеописаниемагазина",
^"ТорговыеЗалы@odata.type":"Collection(StandardODATA.Catalog_Магазины_ТорговыеЗалы_RowType)"
^"ТорговыеЗалы":[
^^{
^^^"LineNumber":"1",
^^^"Название":"Синийзал",
^^^"Площадь":56,
^^^"ДатаОткрытия":"2015-01-01T00:00:00"
^},
^{
^^^"LineNumber":"2",
^^^"Название":"Красныйзал",
^^^"Площадь":56,
^^^"ДатаОткрытия":"2015-06-13T11:45:41"
^}
^]
}
```

> **ВНИМАНИЕ!** Табличную часть следует передавать полностью (все строки) даже в том случае, если требуется изменить данные только в одной строке этой табличной части.

В данном примере guid'value' идентифицирует изменяемый элемент справочника Магазины.

Удаление объекта

Для удаления объекта следует воспользоваться DELETE-запросом с использованием канонического URL сущности.

> **ВНИМАНИЕ!** Пометка на удаление не выполняется, объект удаляется непосредственно.

Оптимистическая блокировка данных

Для оптимистической блокировки данных (т.е. для проверки, что данные не изменились с момента считывания) нужно использовать заголовок If-Match HTTP-запроса, связанного с модификацией (PATCH) или удалением (DELETE) данных. В качестве значения заголовка должно выступать значение свойства DataVersion, которое получено при предварительном чтении сущности.

Механизм работает следующим образом:

- При запросе экземпляра сущности среди прочих возвращается свойство DataVersion. При запросе набора сущностей с каждой из сущностей набора возвращается свойство DataVersion.
- Клиентское приложение должно сохранить у себя это значение версии объекта и затем использовать его в запросах PATCH и DELETE, передавая его в заголовке If-Match.
- Если значение свойства DataVersion в момент исполнения запроса совпадает со значением заголовка If-Match, то происходит запрошенное действие. В противном случае действие не выполняется, а клиенту возвращается HTTP статус 412.

Запись объекта в режиме загрузки данных

Если при записи объекта необходимо эмулировать запись, выполняемую во время работы механизма обмена данными (свойство ОбменДанными.Загрузка = Истина) следует использовать HTTP-заголовок 1C_OData-DataLoadMode со значением true при выполнении соответствующей операции записи.

Проведение и отмена проведения документа

Для проведения документа необходимо выполнить POST-запрос с использованием канонического URL сущности (аналогично запросу GET для получения сущности), указав особым образом сформированный суффикс URL. Суффикс в этом случае будет состоять из команды Post и параметра, указывающего режим проведения документа:
http://host/base/odata/standard.odata/Document_РасходТоваров(guid'value')/Post?PostingModeOperational=false.

Для отмены проведения документа суффикс состоит из команды Unpost без параметров.

17.2.1.13.2. Работа с коллекцией объектов

Для получения набора сущностей какого-либо вида, следует выполнить GET-запрос с использованием URL следующего вида:
http://host/base/odata/standard.odata/ExchangePlan_ОбменДанными.

Если необходимо установить отбор на получаемый список, его можно выполнить с помощью параметра $filter (см. здесь), который указывается в URL:

```
http://host/base/standard.odata/Document_Накладная?$filter=Приоритетеq1
```

Обращение к виртуальной таблице регистра выглядит следующим образом (на примере виртуальной таблицы `ОстаткиИОбороты` регистра накопления `ТоварныеЗапасы`):

```
http://host/base/odata/standard.odata/AccumulationRegister_ТоварныеЗапасы/BalanceAndTurnovers(StartPeriod=datetime'2014-01-01',EndPeriod=datetime'2014-02-01',Condition='Товар_Keyeqguid'value'')
```

Также для ограничения набора сущностей можно использовать параметры $top и $select (см. здесь).

17.2.1.13.3. Работа с планами обмена

Формирование сообщения обмена

Для формирования сообщения обмена необходимо выполнить POST-запрос с использованием URL следующего вида: http://host/base/odata/standard.odata/SelectChanges?<params>. Где в качестве параметров необходимо указать следующее:

- Параметр DataExchangePoint – должен содержать канонический URL сущности треуемого элемента плана обмена;
- Параметр MessageNo – должен содержать номер сообщения обмена данными, который будет сформирован в результате данного вызова.

В результате, полный URL для формирования сообщения обмена, будет выглядеть следующим образом: http://host/base/odata/standard.odata/SelectChanges?DataExchangePoint='http://host/base/odata/standard.odata/ExchangePlan_ОбменДанными(guid'value')'&MessageNo=34.

В результате будет получен список изменений, которые необходимо передать в другой узел, в виде потока atom-feed. Каждый элемент будет представлен в формате atom-entry, а удаленные элементы будет представлены в формате atom-deleted-entry (RFC 6721, http://tools.ietf.org/html/rfc6721, на английском языке).

Уведомление о получении изменений

Для уведомления сервера о том, что сообщение обмена успешно получено, необходимо выполнить POST-запрос с использованием URL следующего вида: http://host/base/odata/standard.odata/ NotifyChangesReceived?<params>. Где в качестве параметров необходимо указать следующее:

- Параметр DataExchangePoint – должен содержать канонический URL сущности требуемого элемента плана обмена;
- Параметр MessageNo – должен содержать номер сообщения обмена данными, подтверждение получения которого необходимо зафиксировать.

В результате, полный URL для подтверждения получения сообщения обмена, будет выглядеть следующим образом: http://host/base/odata/standard.odata/NotifyChangesReceived?DataExchangePoint='http://host/base/odata/standard.odata/ExchangePlan_ОбменДанными(guid'value')'&MessageNo=34.

17.2.2. HTTP-сервисы

17.2.2.1. Общая информация

Помимо стандартного интерфейса OData (см. здесь), система «1С:Предприятие» позволяет прикладному разработчику реализовать собственные HTTP-сервисы. Создание HTTP-сервисов позволяет расширить (по сравнению со стандартным интерфейсом OData) или создать (если стандартный интерфейса OData не используется) некий прикладной интерфейс, который будет доступен с помощью HTTP-запросов. Общее описание протокола HTTP 1.1 приведено по ссылке: http://tools.ietf.org/html/rfc2616 (на английском языке).

HTTP-сервисы (REST-сервисы) являются аналогами SOAP-сервисов, которые в «1С:Предприятии» представлены Web-сервисами (см. здесь). SOAP-сервис – это набор параметризованных действий, причем перечень действий не ограничен. Набор действий определяется во время разработки сервиса. HTTP-сервис – это ограниченный набор действий (ограничен списком HTTP-методов) и разветвленный набор ресурсов. Состав ресурсов определяем в процессе разработки сервиса. Набор ресурсов и определяет функциональные возможности сервиса. При разработке SOAP-сервисов также определяются структуры данных, передаваемые между клиентом и сервером, а при разработке HTTP-сервиса – структуры передаваемых данных жестко не определяются.

Обращение к HTTP-сервису выполняется по некоторому URL. В качестве примера можно привести HTTP-сервис, который при обращении по одному URL возвращает список каких-либо документов (например, расходных накладных), а при обращении по другому URL будет возвращена конкретная накладная. Действие, которое следует выполнить серверу, определяется тем, какой HTTP-метод адресован серверу. Например, если на сервер отправляется GET-запрос, то выполняется получение каких-либо данных, DELETE-запрос приводит к удалению каких-либо данных и т. д.

При реализации HTTP-сервиса прикладной разработчик определяет следующие элементы сервиса:

1. Определяет базовую часть URL, по которому будет выполняться обращение к сервису.
2. Определяет состав предоставляемой функциональности и структуру ресурсов, на которые эта функциональность будет отображаться.
3. Определяет действия (HTTP-методы), которые можно будет выполнить при обращении к тому или иному ресурсу.
4. Для каждого выбранного действия реализуется специальный метод, написанный на встроенном языке системы «1С:Предприятие», который и реализует необходимую функциональность. Метод расположен в специальном модуле, связанным с HTTP-сервисом.

После реализации HTTP-сервиса, необходимо опубликовать его на веб-сервере с помощью стандартного механизма публикации.

При обращении к HTTP-сервису формируется URL, который выглядит следующим образом: http://host/base/hs/корневойURL/относительныйURL. Более подробно рассмотрим составные части адреса:

- http://host/base – обычный URL, по которому выполняется доступ, например, к информационной базе с помощью веб-клиента. При наличии разделителей, не поддерживается указание значений разделителей с помощью параметра Z командной строки

запуска клиентского приложения.

- hs – признак того, что выполняется обращение к HTTP-сервису (в отличие от ws, который определяет доступ к Web-сервису).
- корневойURL – имя ресурса, которое определяет группу ресурсов, связанных общим смыслом. Задается в свойстве объекта HTTP-сервис.
- относительныйURL – определяет ресурс, к которому будет выполняться обращение. Относительный URL, указанный в запросе, будет использован для определения конкретного ресурса, к которому выполнялось обращение. Правило сопоставления задается в объекте Шаблон URL.

17.2.2.2. Разработка HTTP-сервиса

Создание HTTP-сервисов выполняется в ветке Общие – HTTP-сервисы дерева объектов конфигурации. Вначале создается собственно сервис, затем в каждом сервисе необходимо создать определенный шаблон фрагмента URL (группу ресурсов), и затем для каждой группы ресурсов необходимо определить HTTP-метод, который будет реализован для данной группы. Также надо понимать, что один сервис может включать в себя более одного шаблона URL (отношение «один ко многим»). Фактически, шаблон URL определяет группу ресурсов, обрабатываемых по единым правилам.

17.2.2.2.1. Редактирование HTTP-сервиса

При создании HTTP-сервиса следует обратить внимание на свойство Корневой URL. Определяет группу ресурсов, объединенных общим смыслом. Например, если необходимо создать несколько ресурсов, которые работают с заказами, то корневой URL в этом случае может выглядеть как order. Тогда начало URL при обращении к такому сервису будет выглядеть следующим образом: http://host/base/hs/order.

17.2.2.2.2. Редактирование шаблона URL

При создании шаблона URL описываются возможные адреса ресурсов, которые можно будет использовать для обращения к HTTP-сервису. Собственно шаблон задается в свойстве Шаблон. При создании шаблона можно использовать следующие символы:

- Любые символы, допустимые в идентификаторах языка «1С:Предприятие».
- Символ «/»;
- Символы «{}» с непустым текстом между ними;
- Символ *.

Например, шаблон может выглядить следующим образом:

```
/query
/documents/{id}/props/{PropertyName}/*
```

При описании шаблона будут использовать следующие термины:

- Сегмент – часть URL между двумя символами «/» Например, props, из примера выше, является сегментом.
- Параметризованный сегмент – сегмент, заключенный в фигурные скобки («{сегмент}»). Фактически, параметризованный сегмент описывает переменную с указанным именем, к которой можно получить доступ из встроенного языка.

Сегмент (не параметризованный) из шаблона должен быть перенесен в URL дословно. На месте параметризованного сегмента может располагаться любое значение, допустимое в URL. Символ «*» может находиться только в конце шаблона. На месте символа «*» может находиться любое количество сегментов, включая отсутствие сегмента. В одном шаблоне не может встречаться параметризованных сегментов с одинаковым именем. Имя параметризованного сегмента может состоять только из букв, цифр и символа «_».

Шаблон вида /documents/*/number не является допустимым. Если задан шаблон вида /documents/{kind}/{number}, то являются допустимыми следующие URL сервиса:

```
/documents/trade/13
/documents/12/test
```

При получении запроса система пытается определить, какой шаблон следует использовать. Для этого выполняется попытка сопоставить текст входящего запроса с заданными шаблонами. Сопоставление выполняется по следующим правилам:

1. Все строки сравниваются регистрозависимо;
2. Точное совпадение всегда более приоритетно, чем соответствие шаблону с параметризованными сегментами или символом «*»;
3. Символ «*» сопоставляется с любым количеством сегментов (включая отсутствие сегмента);
4. Параметризованный сегмент имеет приоритет над символом «*»;
5. Если переданному запросу не удалось однозначно сопоставить ни один шаблон – будет сгенерирована ошибка.

Данные правила поясняют следующие примеры:

1. При сопоставлении URL /Накладная с шаблонами /Накладная/ и /{EntityName}/ для использования будет выбран первый шаблон;
2. Шаблон /Document/*/{Num} не является допустимым;
3. URL /Накладная/500/Строки/2/Товар может быть сопоставлен шаблону /Накладная/*;
4. При сопоставлении URL /Накладная/500 с шаблонами /Document/{Num} и /{Document}/*, для использования будет выбран первый шаблон;
5. При сопоставлении URL /Накладная/500/ шаблонам {Document}/{Num} и /{Document}/{Number} произойдет ошибка, т. к. не удалось однозначно выбрать шаблон.

17.2.2.2.3. Редактирование метода

Для созданного шаблона необходимо определить HTTP-методы, которые могут быть использованы при работе с ресурсом, совпадающим с шаблоном. Для этого необходимо создать нужное количество объектов Метод, подчиненных объекту Шаблон URL. В каждом методе необходимо указать свойство HTTP-метод и создать обработчик. Имя обработчика автоматически формируется из имени шаблона URL и собственно имени метода. Так, если в шаблоне URL указано имя ПримерДляДокументации, и для метода указано имя GET, то для метода встроенного языка, с помощью которого будет обрабатываться HTTP-запрос, использовано имя обработчика `ПримерДляДокументацииGET()`.

СОВЕТ. Рекомендуется называть объекты HTTP-метод именем обслуживаемого метода на английском языке.

Имеется возможность создать обработчики для следующих методов:

- GET;
- POST;
- PUT;
- DELETE;
- PATCH;
- MERGE;
- CONNECT;
- OPTIONS;
- TRACE;
- PROPFIND;
- PROPPATCH;
- MKCOL;
- COPY;
- MOVE;
- LOCK;
- UNLOCK.

Если создается обработчик для метода Любой, то обработчик этого метода будет вызван для всех HTTP-методов (из приведенного выше списка), которые явно не описаны в том шаблоне, который «обрабатывает» входящий HTTP-запрос.

После получения необходимого шаблона, будет предпринята попытка вызвать метод, обслуживающий нужный HTTP-запрос (GET, POST и т. д.). Если обработчик для необходимого запроса не был обнаружен, вызывающей стороне будет возвращена ошибка 405. Если в процессе исполнения метода обнаружена ошибка встроенного языка – вызывающей стороне будет возвращена ошибка 500, а в тело ответа будет помещен текст исключения.

При формировании ответа параметр Content-Length может быть не задан. В этом случае он будет определен автоматически.

17.2.2.2.4. Входные и выходные данные обработчика метода

Обработчиком того или иного HTTP-метода является функция, которая получает на вход один параметр типа `HTTPСервисЗапрос` и должна вернуть объект типа `HTTPСервисОтвет`.

Входной параметр полностью описывает запрос, поступивший в систему. С помощью объекта можно определить:

- Какой HTTP-метод используется в запросе (свойство `HTTPМетод`);
- На какой URL поступил запрос (свойство `БазовыйURL`) – часть полного URL запроса, не включающая относительный URL (см. здесь) и параметры запроса;
- Относительный URL (свойство `ОтносительныйURL`) – часть полного URL запроса, которая использовалась для определения группы ресурсов и правил обработки входящего HTTP-запроса;
- Какие заголовки (свойство `Заголовки`) содержит HTTP-запроса;
- Какие параметризованные сегменты (свойство `ПараметрыURL`) выделены из входящего HTTP-запроса и их значения.
- Какие параметры (свойство `ПараметрыЗапроса`) перечислены в URL HTTP-запроса после символа "?" и их значения.

Также существует возможность получать тело запроса в виде двоичных данных или строки (в зависимости от передаваемой информации).

Таким образом, прикладному разработчику не требуется самостоятельно «разбираться» с входящим запросом. С помощью примера, приведенного далее в этой главе (см. здесь), можно смотреть, каким образом входящий HTTP-запрос распределяется по свойствам объекта `HTTPСервисЗапрос`.

После того, как прикладное решение выполнило все действия, которые определены для данного ресурса и HTTP-метода, необходимо сформировать ответ сервиса. Для этого необходимо создать объект типа `HTTPСервисОтвет`. В зависимости от результата обработки, следует указать свойство `КодСостояния`, которое описывает стандартный код возврата HTTP (http://tools.ietf.org/html/rfc2616#section-10, на английском языке). Затем, при необходимости, следует сформировать тело ответа HTTP-сервиса. Тело может быть установлено как двоичные данные, как строка или с помощью указания имени файла, откуда будет загружено тело ответа. Если тип возвращаемого функцией (которая реализует реакцию на HTTP-метод) значения отличается от `HTTPСервисОтвет`, то клиенту будет возвращена 500 ошибка.

17.2.2.3. Примеры реализации HTTP-сервисов

ПРИМЕЧАНИЕ. Примеры, приведенные ниже, не являются законченным. Они предназначены для демонстрации некоторых возможностей HTTP-сервисов.

17.2.2.3.1. Отображение входящего запроса

В качестве примера реализуем HTTP-сервис, который будет возвращать в качестве ответа все параметры входящего запроса.

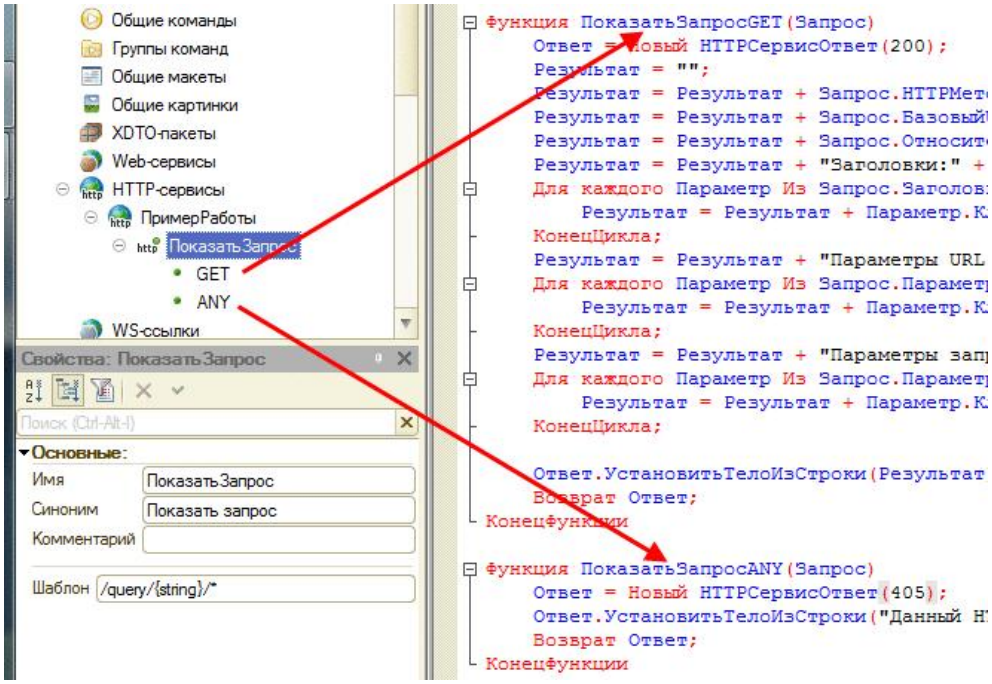

Рис. 459. Тестовый пример

Для этого нужно создать HTTP-сервис ПримерРаботы, указав для него корневой URL example. Затем следует для сервиса создать шаблон URL ПоказатьЗапрос с шаблоном /query/{string}/*. Теперь необходимо в шаблон ПоказатьЗапрос добавить следующие методы:

- Метод GET для HTTP-запроса GET;
- Метод ANY для HTTP-запроса Любой.

Обработчики должны выглядеть следующим образом:

```
ФункцияПоказатьЗапросGET(Запрос)
^Ответ=НовыйHTTPСервисОтвет(200);
^Результат="";
^Результат=Результат+Запрос.HTTPМетод+Символы.ПС;
^Результат=Результат+Запрос.БазовыйURL+Символы.ПС;
^Результат=Результат+Запрос.ОтносительныйURL+Символы.ПС;
^Результат=Результат+"Заголовки:"+Символы.ПС;
^ДлякаждогоПараметрИзЗапрос.ЗаголовкиЦикл
^^Результат=Результат+Параметр.Ключ+":"+Параметр.Значение+Символы.ПС;
^КонецЦикла;
^Результат=Результат+"ПараметрыURL:"+Символы.ПС;
^ДлякаждогоПараметрИзЗапрос.ПараметрыURLЦикл
^^Результат=Результат+Параметр.Ключ+":"+Параметр.Значение+Символы.ПС;
^КонецЦикла;
^Результат=Результат+"Параметрызапроса:"+Символы.ПС;
^ДлякаждогоПараметрИзЗапрос.ПараметрыЗапросаЦикл
^^Результат=Результат+Параметр.Ключ+":"+Параметр.Значение+Символы.ПС;
^КонецЦикла;
^
^Ответ.УстановитьТелоИзСтроки(Результат);
^ВозвратОтвет;
КонецФункции
ФункцияПоказатьЗапросANY(Запрос)
^Ответ=НовыйHTTPСервисОтвет(405);
^Ответ.УстановитьТелоИзСтроки("ДанныйHTTP-сервиснеподдерживаетметод"+Запрос.HTTPМетод);
^ВозвратОтвет;
КонецФункции
```

После публикации HTTP-сервисов, описанной в книге 1С:Предприятие 8.3. "Руководство администратора", сервис должен возвращать полное «описание» поступившего на вход запроса.

При использовании следующего запроса: http://localhost/httpservice/hs/example/query/value/another?p1=1&p2=2 будет получен следующий ответ (значение некоторых заголовков зависит от используемого веб-браузера):

```
GET
http://localhost/httpservice/hs/example
/query/value/another
Заголовки:
Accept-Encoding:gzip,deflate,sdch
Connection:keep-alive
```

```
Accept-Language:ru-RU,ru;q=0.8,en-US;q=0.6,en;q=0.4
DNT:1
Accept:text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Content-Length:0
User-Agent:Mozilla/5.0(WindowsNT6.1;WOW64)AppleWebKit/537.36(KHTML,likeGecko)Chrome/32.0.1700.107Safari/537.36
Host:localhost
ПараметрыURL:
*:/another
string:value
Параметрызапроса:
p1:1
p2:2
```

Рассмотрим, каким образом был разобран запрос. В примере было указано, что корневой URL сервиса – example, шаблон URL – /query/{string}/* и был сформирован обработчик HTTP-запроса GET.

В результате видно, что система определила следующие параметры URL:

- Параметр * со значением /another;
- Параметр string со значением value.

Это полностью соответствует указанному шаблону URL. Если шаблон URL установить в значение /query/{string}, то демонстрационный запрос закончится сообщением об ошибке 404, т. к. такой шаблон не будет допускать наличие в URL информации, после параметризованного сегмента.

Также в обработчике метода будут переданы все параметры запроса, которые были указаны в URL.

Любой другой вид запроса (отличный от GET) будет приводить к ошибке 405 с указанием, какой именно запрос не поддерживается сервисом.

17.2.2.3.2. Работа с документами

Данный пример является некоторым аналогом Web-сервиса, который должен по переданному номеру расходной накладной возвращать состав ее табличной части (см. здесь).

Данный пример будет предоставлять возможность получить список «документов» и получить «документ» по номеру. Также будет реализована реакция на неверный номер «документа». Для упрощения примера реальных объектов типа Документ в примере создано не будет.

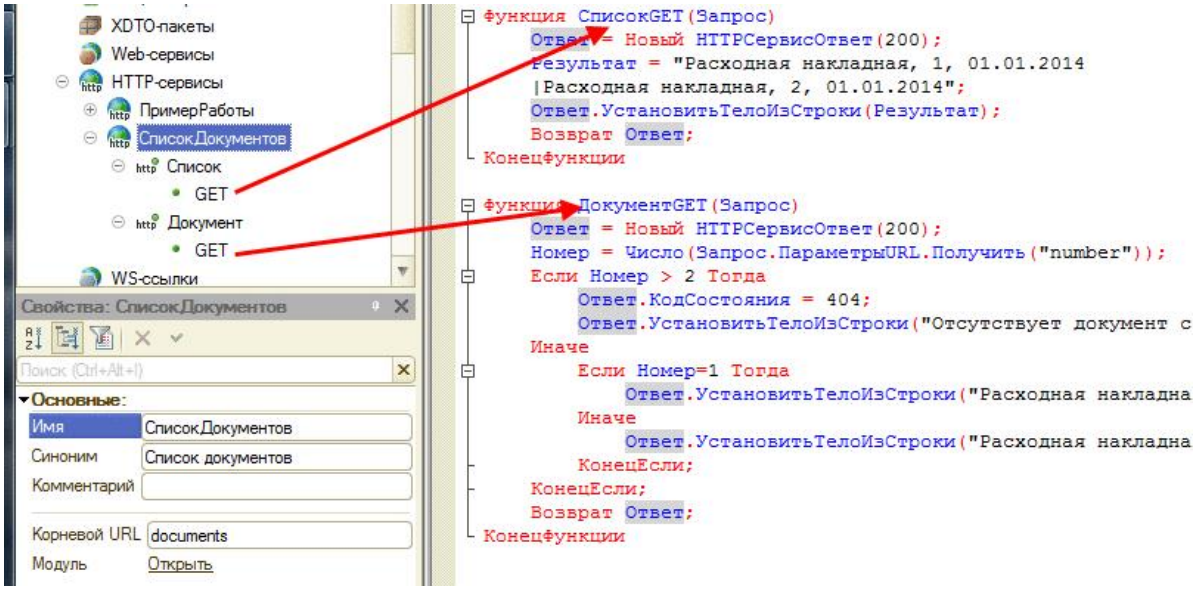

Рис. 460. Тестовый пример

Для этого нужно создать HTTP-сервис СписокДокументов, указав для него корневой URL documents. Затем следует для сервиса создать шаблон URL Список с шаблоном /* и шаблон URL Документ с шаблоном /{number}. Для каждого шаблона URL следует добавить реализацию метода GET.

Обработчики должны выглядеть следующим образом:

```
ФункцияСписокGET(Запрос)
^Ответ=НовыйHTTPСервисОтвет(200);
^Результат="Расходнаянакладная,1,01.01.2014
^|Расходнаянакладная,2,01.01.2014";
^Ответ.УстановитьТелоИзСтроки(Результат);
^ВозвратОтвет;
КонецФункции
ФункцияДокументGET(Запрос)
^Ответ=НовыйHTTPСервисОтвет(200);
^Номер=Число(Запрос.ПараметрыURL.Получить("number"));
^ЕслиНомер>2Тогда
^^Ответ.КодСостояния=404;
^^Ответ.УстановитьТелоИзСтроки("Отсутствуетдокументсномером:"+Номер);
^Иначе
^^ЕслиНомер=1Тогда
^^^Ответ.УстановитьТелоИзСтроки("Расходнаянакладная,1,01.01.2014");
^^Иначе
^^^Ответ.УстановитьТелоИзСтроки("Расходнаянакладная,2,01.01.2014");
```

```
^^КонецЕсли;
^КонецЕсли;
^ВозвратОтвет;
КонецФункции
```

После публикации HTTP-сервисов, описанной в книге 1С:Предприятие 8.3. "Руководство администратора", сервис должен возвращать некоторый набор данных о документах, «хранящихся» в системе.

Рассмотрим, каким образом работает созданный сервис:

- Запрос http://localhost/httpservice/hs/documents вернет список «документов»:

```
Расходнаянакладная,1,01.01.2014
Расходнаянакладная,2,01.01.2014
```

- Запрос вида http://localhost/httpservice/hs/documents/get/list вернет список «документов», т. к. этот запрос соответствует шаблону Список и не соответствует шаблону Документ.

- Запрос вида http://localhost/httpservice/hs/documents/list вернет ошибку времени исполнения, т. к. приведенный запрос соответствует шаблону Документ, но в функции `ДокументGET()` нет проверки на то, что переданный номер является числом.

- Запрос вида http://localhost/httpservice/hs/documents/2 вернет «документ» с номером 2:

```
Расходнаянакладная,2,01.01.2014
```

- Запрос вида http://localhost/httpservice/hs/documents/3 вернет ошибку:

```
Отсутствуетдокументсномером:3
```

Обработка корректности номера документа явно реализована в обработчике метода.

17.2.2.4. Установка значений разделителей

Значения разделителей можно устанавливать только следующими способами:

- С помощью фрагментов URL при обращении к REST-сервису;

- С помощью файла описания публикации default.vrd.

Указанием разделителей с помощью параметры командной строки Z не поддерживается.

17.2.2.5. Публикация HTTP-сервиса

Публикация HTTP-сервисов выполняется с помощью диалога публикации на веб-сервере (Администрирование – Публикация на веб-сервере) и описана в книге 1С:Предприятие 8.3. "Руководство администратора".

17.3. Повторное использование сеансов интернет-сервисов

17.3.1. Общая информация

Устройство системы «1С:Предприятие» таково, что вызов интернет-сервиса (Web-сервис, HTTP-сервис, стандартный интерфейс OData) и запуск клиентского приложения выглядит примерно одинаково – система создает сеанс работы с информационной базой (подробнее см. здесь), в котором и выполняются все необходимые действия. При завершении вызова интернет-сервиса (и при завершении работы клиентского приложения) сеанс завершается, а при повторном вызове процесс повторяется заново. Очевидным недостатком такого поведения является существенное время, которое тратит система на создание сеанса (включая выполнение всех обработчиков событий и загрузку собственно конфигурации). Особенно это заметно на «больших» прикладных решениях, которые содержат много различных объектов конфигурации. Для клиентского приложения время создания сеанса может быть несущественным (на фоне общего времени непрерывной работы клиентского приложения), а для интернет-сервисов это время может привносить существенные задержки в общее время вызова сервиса. Система предоставляет возможность управлять необходимостью создания сеанса при каждом вызове интернет-сервиса:

- Не использовать повторное использование сеансов. При каждом обращении к интернет-сервису будет создаваться новый сеанс.

- Включить повторное использование сеансов. В этом случае создание и завершение сеансов регулируется вызывающей стороной с помощью заголовков HTTP-запроса.

- Включить автоматическое повторное использование сеансов. В этом случае системой организуется пул сеансов. Каждый сеанс в пуле характеризуется некоторым набором параметров. Если сеанс в пуле свободен и характеризуется тем же набором параметров, что и запрос интернет-сервиса, то запросу сразу будет предоставлен существующий сеанс.

Возможность повторного использования сеансов настраивается в нескольких местах:

- в свойствах объектов конфигурации Web-сервис и HTTP-сервис (вид повторного использования и время жизни сеанса);

- в файле публикации default.vrd (параметры пула сеансов).

Следует помнить, что значения, указанные в файле default.vrd, имеют приоритет над значениями, которые заданы в свойствах объектов конфигурации.

17.3.2. Виды повторного использования сеансов

Не использовать

В этом случае каждый запрос интернет-сервиса приведет к созданию нового сеанса.

Использовать

В этом случае временем жизни и характером повторного использования сеанса управляет непосредственно клиент интернет-сервиса. В этом случае для управления созданием нового сеанса используются специальные заголовки HTTP-запроса к интернет-сервису.

Использовать автоматически

Пул сеансов ведется в разрезе следующих параметров:

- тип сервиса;
- имя сервиса;
- пользователь;
- пароль;
- набор разделителей;
- безопасный режим.

При обнаружении входящего запроса к интернет-сервису, система анализирует текущий пул сеансов и ищет сеанс с параметрами, в точности соответствующими входящему запросу. Если такой сеанс обнаруживается и он не используется запросом от другого клиента интернет-сервиса, входящий запрос будет обслуживаться найденным сеансом. Если свободного сеанса не найдено – выполняется попытка создать новый сеанс. Если при попытке создания нового сеанса будет превышен размер пула сеансов – входящий запрос будет ожидать некоторое время (таймаут пула). Если по истечению таймаута пула свободный сеанс не обнаруживается – интернет-запрос будет завершен с ошибкой 406 Not Acceptable. Свободный сеанс будет уничтожен системой автоматически после истечения времени жизни свободного сеанса.

Настройки автоматического пула сеансов действуют в рамках публикации. Таким образом, если для некоторой информационной базы созданы несколько публикаций, то при вызове интернет-сервиса используется настройка пула той публикации, через которую выполняется вызов.

Рекомендуется подбирать такой размер пула сеансов, чтобы при максимальной загрузке в пуле оставалось несколько свободных сеансов (т.е. выбирать размер пула с небольшим запасом). В противном случае при пиковой нагрузке некоторые вызовы будут завершаться с ошибками. Система «1С:Предприятие» не поддерживает автоматическое определение необходимого размера пула сеансов.

17.3.3. Настройки повторного использования сеансов

17.3.3.1. Свойства интернет-сервисов

В палитре свойств интернет-сервисов (Web-сервис, HTTP-сервис), включая собственные интернет-сервисы расширений конфигурации, присутствуют два свойства, описывающие повторное использование сеансов:

- Свойство Повторное использование сеансов – указывает вид повторного использования сеансов для настраиваемого сервиса (подробнее см. здесь).
- Свойство Время жизни сеанса:
  - описывает, через какой промежуток времени «свободный» сеанс будет удален из пула сеансов (в случае автоматического повторного использования сеансов);
  - описывает, через какое время бездействия сеанс будет принудительно завершен системой при ручном повторном использовании сеансов.

  Если данное свойство установлено в значение 0, то это означает, что сеансов повторно не используются. Такая установка эквивалентна установке свойства Повторное использование сеансов в значение Не использовать.

Параметры пула сеансов для автоматического повторного использования сеансов настраиваются с помощью файла default.vrd (подробнее см. здесь).

17.3.3.2. На стороне клиента

Если для интернет-сервиса в качестве значения свойства Повторное использование сеансов, то в этом случае управлением временем жизни сеанса занимается клиент интернет-сервиса. В качестве управляющего элемента выступает заголовок HTTP-запроса `IBSession`. Этот заголовок может принимать два значения:

- `start` – в этом случае система «1С:Предприятие» создает новый сеанс, выполняет аутентификацию, устанавливает разделители, выполняются все необходимые обработчики событий. Если система не может создать новый сеанс, то клиент получит ошибку 406 Not Acceptable. Если создание сеанса выполнено успешно, в HTTP-ответ помещается директива установить cookie `ibsession` с идентификатором созданного сеанса: Set-Cookie: ibsession=<ID сеанса>.

  При необходимости использовать ранее созданный сеанс, необходимо в HTTP-запросе к системе «1С:Предприятие» указать идентификатор ранее созданного сеанса: Cookie: ibsession=<ID сеанса>. Если в запросе указывается идентификатор сеанса, который ранее не создавался или был завершен, клиент получает ошибку 400 Bad Request.

  Если в HTTP-запрос не содержит заголовка `IBSession`, то сеанс создается и завершается при каждом вызове интернет-сервиса.

  Если в процессе использования сеанса в HTTP-запросе изменяются значения разделителей или безопасный режим сеанса, то новые параметры игнорируются и сеанс будет использовать те значения, которые были указаны при старте сеанса.

- `finish` – в этом случае система «1С:Предприятие» завершает сеанс, который указан в запросе, одновременно с командой завершения сеанса: Cookie: ibsession=<ID сеанса>. Завершение сеанса произойдет автоматически, если в этом сеансе не выполнялось никаких действий за время жизни сеанса.

Пул сеансов в этом случае не используется.

17.3.3.3. В файле default.vrd

Описание параметров файла default.vrd, которые управляют повторным использованием сеансов, см. здесь.

17.3.3.4. Для стандартного интерфейса OData

Режим и параметры повторного использования сеансов для стандартного интерфейса OData устанавливаются системой «1С:Предприятие» при публикации стандартного интерфейса OData на веб-сервере. При этом по умолчанию используются следующие значения:

- Повторное использование сеансов – включено автоматическое использование.
- Время жизни сеанса – 20 секунд.