**WCF Data Services:** протокол Open Data Protocol, возможности предоставляемые OData-интерфейсом, порядок разработки WCF Data Services, применение WCF Data Services.

WCF Data Service – реализация стандарта протокола open data protocol.

Open data protocol – открытый web-протокол; позволяет выполнять операции с ресурсами и получать ответы в формате XML, JSON. Поддерживается OASIS.

Ключевые возможности OData:

1. Получения списка сущностей
2. Получения записей по каждой сущности
3. Получения сущности по ID
4. Фильтрация сущностей по признаку
5. Получение отдельных атрибутов сущности
6. Создание сущности
7. Редактирование сущности
8. Удаление сущности

**Архитектура:**

Клиент (OData client Library => ODataClient) => ODataProtocol => Data Sources (ODataService (OData data modal) => Different Data Source)

Порядок разработки:

1. Создаем бд. Добавляем таблицу и заполняем данными
2. В Visual Studio создаем элемент, выбираем Модель ADO .NET EDM, создаем из базы данных, используем проверку подлинности SQL Server и задаем имя которое будет указано в WebConfig для нашей модели. Выбираем таблицы которые мы портируем и выбираем готово.
3. У нас создаться edmx файл который будет классом, который наследуется от DBContext, внутри него будут методы которые возвращают множество наших сущностей
4. Добавляем новый элемент и выбираем WCF Data Service. Расширение файла будет svc. Его нужно будет унаследовать от нашей модели.
5. После этого мы можем проверить работоспособность в браузере нашего приложения. Мы можем прямо в url писать запросы и на странице получать ответ. localhost:port/DWS.svc/Student(1)/Name
6. Добавим клиента. Для этого нужно добавить ссылку на службу и указать url нашего сервиса, после добавить
7. Получаем экземпляр через референс сервиса wds = new ServiceReference1.WDSEntities

В браузере чтобы получить JSON нужно указать в url $format=json

localhost:port/WCFDataService1.svc/Students/$format=json

**WCF Syndication Services:** стандарты ATOM, RSS, порядок разработки WCF Syndication Service.

WCF Syndication Service — это сервис, реализованный с использованием Windows Communication Foundation (WCF), который предоставляет возможность работы с синдикацией контента, например,Atom и RSS ленты.

RSS – семейство XML-форматов, предназначенного для описания новостных лент, анонсов и статей.

application/rss+xml  
Последняя версия 2.0 (2007)

Интернет-ресурс в формате rss называется rss-каналом, rss-лентой, rss-филдом.

RSS-агрегатор – web-приложение, которое автоматически с заданным интервалом времени проверяют, указанные в настройках ресурсы и экспортируют в формате rss или atom информацию

RSS элементы:

Required channel elements: title, link, description

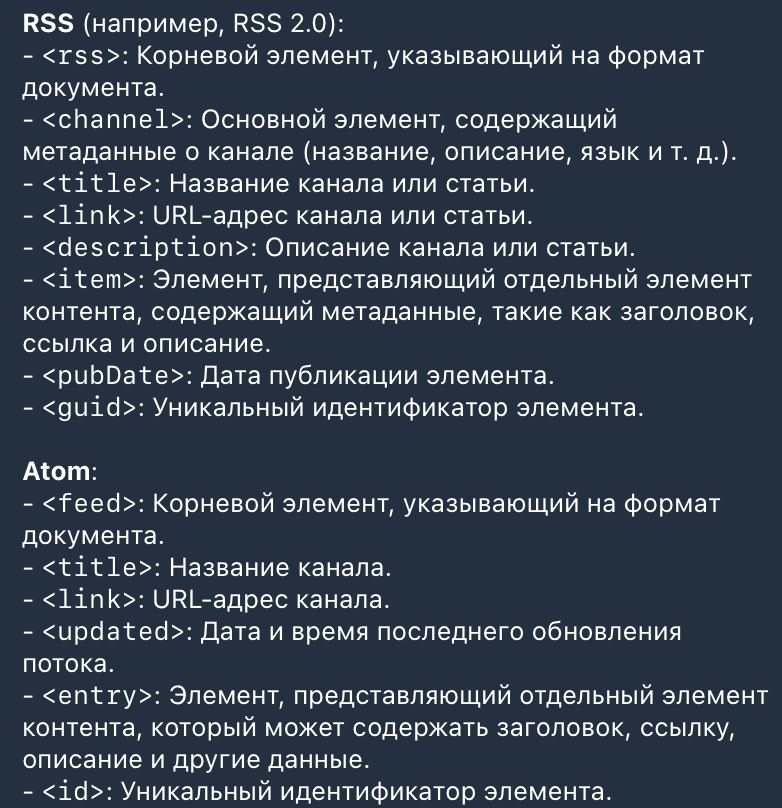
Optional channel elements: language, copyrigth, webmaster, pubDate, managingEditor

ATOM – формат описания веб ресурсов и протокола для их публикации

Последняя версия 1.0

application/atom+xml

ATOM более универсальный и чаще используется. Следует применять при разработке новых приложений.

****

**Порядок разработки:**

1. В созданном проект добавить новый элемент из WCF – Библиотека службы публикации. Сгенерируется класс, который наследуется от интерфейса IFeed. В нем идет создания объекта, указания значений атрибутам и два метода для создания и получения массива новостей
2. В Config нужно указать адрес на наши методы выставить бидинг и указать baseAddress.

localhost:port/SyndicationService/?wsdl–описание   
localhost:port/SyndicationService/feed – ответ нашей апихи

через $format=atom || rss можно указать формат

1. Далее захостовать проект и создать клиента прокинув сервис.

**JSON-RPC:**  определение JSON-RPC-сервиса, форматы запросов и ответов, обработка ошибок, пакеты запросов, реализация JSON-RPC на платформе Web API.

Что такое JSON-RPC?

JSON-RPC – протокол удаленного вызова процедур, использующий формат JSON для передачи сообщений.

Последняя версия 2.0.

Транспорт HTTP/HTTPS или TCP Socket.

Структура запроса  
{  
 “jsonrpc”: “2.0” - Версия протокола

“method”: “get” - Имя вызываемого метода

“params”: [] || {} - Параметры, могут быть переданы массивом или объектом

“id”: 1 - Индификатор запроса, чтобы сопоставить ответ с запросом  
}

Виды JSON-RPC запросов:

1. Синхронные запросы: клиент ожидает ответ сразу, и после ответа продолжает выполнение.
2. Асинхронный запрос: Клиент отправляет запрос и продолжает выполнение, не дожидаясь ответа.
3. Уведомление: запросы без поля id, которые используются просто для отправки данных на сервер без необходимости в ответе

Структура уведомления

{  
 “jsonrpc”: “2.0”,

“method”: “имя вызываемого метода”,

“params”: []  
}

Структура ответа

{

“jsonrpc”: “2.0”,

“result”: data: {},  
 “error”: {

code: 123,

message: ‘Message’,

data: { Доп инфа },  
 }

“id”: 1,  
}

Коды ошибок  
  
-32600 – Неправильный JSON (ошибка синтаксиса)

-32601 – Метод не найден

-32602 – Неверные параметры

-32603 – Внутренняя ошибка сервера

-32000 – Ошибка запроа  
  
  
Можно также отправлять пакеты запросов и пакеты ответов, это будет выглядеть в виде массива, где элементы являются объектами запроса и объектами ответа.