Aplicações Delphi REST utilizando ODATA

Wagner R. Landgraf (TMS Software)

Embarcadero Conference

embarcadero

REST

- Sigla para REpresentational State Transfer.
- Não é uma especificação
- É uma técnica, conceito de arquitetura de software.
- Usa conceito de *resources*.
- *Stateless*, c*acheable*, escalável
- Simples – usa protocolo HTTP – métodos e URL

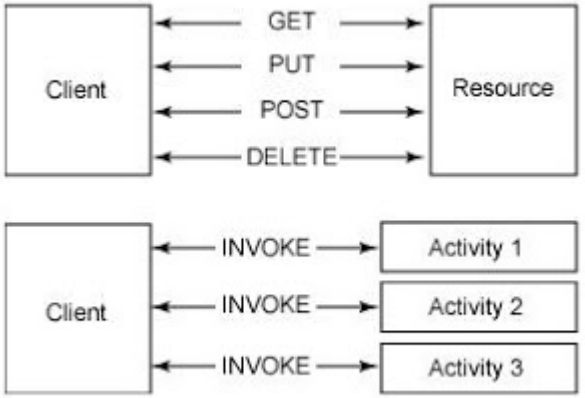
Comparação de conceitos de arquitetura

Orientado a atividades:

Cliente executa métodos no servidor através de um protocolo específico (SOAP, por exemplo):

GetCustomer(1)

ChangeCustomerName(1, “Jack”)



Orientado a recursos:

Cliente usa protocolo HTTP para manipular recursos no servidor (REST):

GET http://server/resources/customer/1 (representação do cliente é retornada pelo servidor)

PUT http://server/resources/customer/1 (cliente fornece a nova representação do cliente na requisição http)

Criando/consumindo uma API REST

- Há padrão para formato da URL dos resources? Não.
  - » A URL pode ser definida livremente, basta representar um resource
- Há padrão para a representação dos resources? Não.
  - » Pode ser XML, JSON, Texto, e ainda assim ser formatado como quiser
- Há padrão para parâmetros e filtros? Não.
- Padrão para paginação, autenticação, errors? Não

OData

- Sigla para “Open Data Protocol”
- Especificação para API WEB para manipulação de dados (CRUD)
- Baseado em HTTP, JSON, AtomPub
- Segue os conceitos de arquitetura REST
- Motivado pela falta de uniformização das API’s REST
- Padroniza o caminho da URL, o formato dos dados, das queries, paginação.
- Fornece metadados das informações

- *Falando nos termos do mundo Delphi, é como se fosse um TDatabase + TDataset via web*

Convenção da URL

```
http://services.odata.org/OData/OData.svc/Category(1)/Products?$top=2&$orderby=name
_____________________________________/ _________________/  _________________/
                    |                          |                   |
            service root URI            resource path       query options
```

Exemplos:

a) http://services.odata.org/OData/OData.svc/Categories
b) http://services.odata.org/OData/OData.svc/Categories(1)
c) http://services.odata.org/OData/OData.svc/Categories(1)/Name
d) http://services.odata.org/OData/OData.svc/Categories(1)/Products
e) http://services.odata.org/OData/OData.svc/Products?$orderby=Rating
f) http://services.odata.org/OData/OData.svc/Products?$top=5
g) http://services.odata.org/OData/OData.svc/Suppliers?$filter=Address/City eq 'Redmond'
h) http://services.odata.org/OData/OData.svc/Products?$filter=Price le 3.5 or Price gt 200

OData

Representação do resource em JSON:


```
{
    "CustomerID": "ALFKI",
    "CompanyName": "Alfreds Futterkiste",
    "Address": {
        "Street": "57 Contoso St",
        "City": "Seattle"
    },
    "Version": "AAAAAAAA+gE=",
    "Orders": {
        "__deferred": {
            "uri": "Customers(\'ALFKI\')/Orders"
        }
    },
    "__metadata": {
        "uri": "Customers(\'ALFKI\')",
        "type": "SampleModel.Customer",
        "etag": "W/\"X\'000000000000FA01\'\"",
        "properties": {
            "Orders": {
                "associationuri": "Customers(\'ALFKI\')/$links/Orders"
            }
        }
    }
}
```

OData

Representação do resource em XML:

```
<?xml version="1.0" encoding="UTF-8"?>
<entry xmlns="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" xml:base="http://host/service.svc/">
   <category term="SampleModel.Customer" scheme="http://schemas.microsoft.com/ado/2007/08/dataservices/scheme" />
   <id>http://host/service.svc/Customers('ALFKI')</id>
   <title type="text" />
   <updated>2008-03-30T21:32:23Z</updated>
   <author>
      <name />
   </author>
   <link rel="edit" title="Customers" href="Customers('ALFKI')" />
   <link rel="http://schemas.microsoft.com/ado/2007/08/dataservices/related/Orders" type="application/atom+xml;type=feed"
title="Orders" href="Customers('ALFKI')/Orders" />
   <link rel="http://schemas.microsoft.com/ado/2007/08/ dataservices/relatedlinks/Orders" type="application/xml"
title="Orders" href="Customers('ALFKI')/$links/Orders" />
   <content type="application/xml">
      <m:properties>
         <d:CustomerID>ALFKI</d:CustomerID>
         <d:CompanyName>Alfreds Futterkiste</d:CompanyName>
         <d:Address>
            <d:Street>57 Contoso St</d:Street>
            <d:City>Seattle</d:City>
         </d:Address>
         <d:Version>AAAAAAAA+gE=</d:Version>
      </m:properties>
   </content>
</entry>
```

OData

Metadados

- Endereço /$metadata contém os metadados
- Exemplo: http://services.odata.org/OData/OData.svc/$metadata
- Fornece informações completas sobre os objetos disponíveis, suas propriedades, tipos de dados, relacionamentos, etc.

Service Document

É o caminho raiz do serviço e traz uma lista de todos os endereços primários (entity sets) disponíveis (fazendo uma analogia com o banco de dados, a lista de tabelas):


```
{
    "odata.metadata": "http://services.odata.org/Northwind/Northwind.svc/$metadata",
    "value": [{
        "name": "Categories",
        "url": "Categories"
    }, {
        "name": "CustomerDemographics",
        "url": "CustomerDemographics"
    }, {
        "name": "Customers",
        "url": "Customers"
    }, {
        "name": "Employees",
        "url": "Employees"
    }, {
        "name": "Order_Details",
        "url": "Order_Details"
    }, {
        "name": "Orders",
        "url": "Orders"
    }, {
    ...
```

OData

Operações CRUD

- Para criar um resource (objeto, registro) usa-se POST no endereço do recurso, passando a representação JSON ou XML.
- Para ler um resource, usa-se GET no endereço, o servidor retorna a representação JSON ou XML
- Para atualizar um resource, usa-se PUT no endereço, fornecendo a representação JSON ou XML. Pode-se usar também MERGE/PATCH e fornecer a representação parcial, incluindo no JSON/XML apenas as propriedades que se deseja alterar
- Para excluir um resource, usa-se DELETE no endereço.

OData

Vantagens

- Não é necessário projetar uma interface API (reinventar a roda)
- Interoperabilidade – suportado em .NET, JavaScript, Java, PHP, etc.
- Facilidade – Diversas frameworks já disponíveis em diversas linguagens, que se comunicam facilmente com servidores no formato Odata
- Conveniência – Diversos outras ferramentas também fazem uso do padrão, como Excel, componentes de grid para JavaScript, WCF Data Services, etc.
- Fácil uso mesmo sem frameworks: HTTP, REST, JSON

OData

Passo-a-passo

Construção de uma aplicação multicamadas em Delphi, com suporte a OData

Usando TMS Aurelius e TMS XData

Especificações

Suporte a dispositivos móveis:

	iOS	Android
TMS Aurelius	Suportado desde a versão 2.1	Suportado na versão 2.3, será lançada
TMS XData	Suportado nas aplicações client	Suportado nas aplicações client

Bancos de dados suportados:

MySQL, Firebird, MS SQL Server, Oracle, Interbase, SQLite, PostgreSQL, DB2, NexusDB, ElevateDB Server, Absolute Database.

Componentes de acesso a banco suportados:

dbExpress, ADO (dbGo), AnyDac, FireDac, UniDAC, SQL-Direct, Direct Oracle Access, FIBPlus, IBObjects, Interbase Express (IBX), Unified Interbase (UIB), AbsoluteDB, ElevateDB, NexusDB e TMS RemoteDB Server.

Obrigado!!!

Wagner R. Landgraf – TMS Software

wagner@tmssoftware.com

http://www.tmssoftware.com

Canais Embarcadero

http://edn.embarcadero.com

http://www.embarcadero.com/br

http://www.facebook.com/DelphiBrasil

http://www.facebook.com/EmbarcaderoBrasil

http://www.embarcadero.com/mvp-directory

http://www.embarcaderobr.com.br/treinamentos/