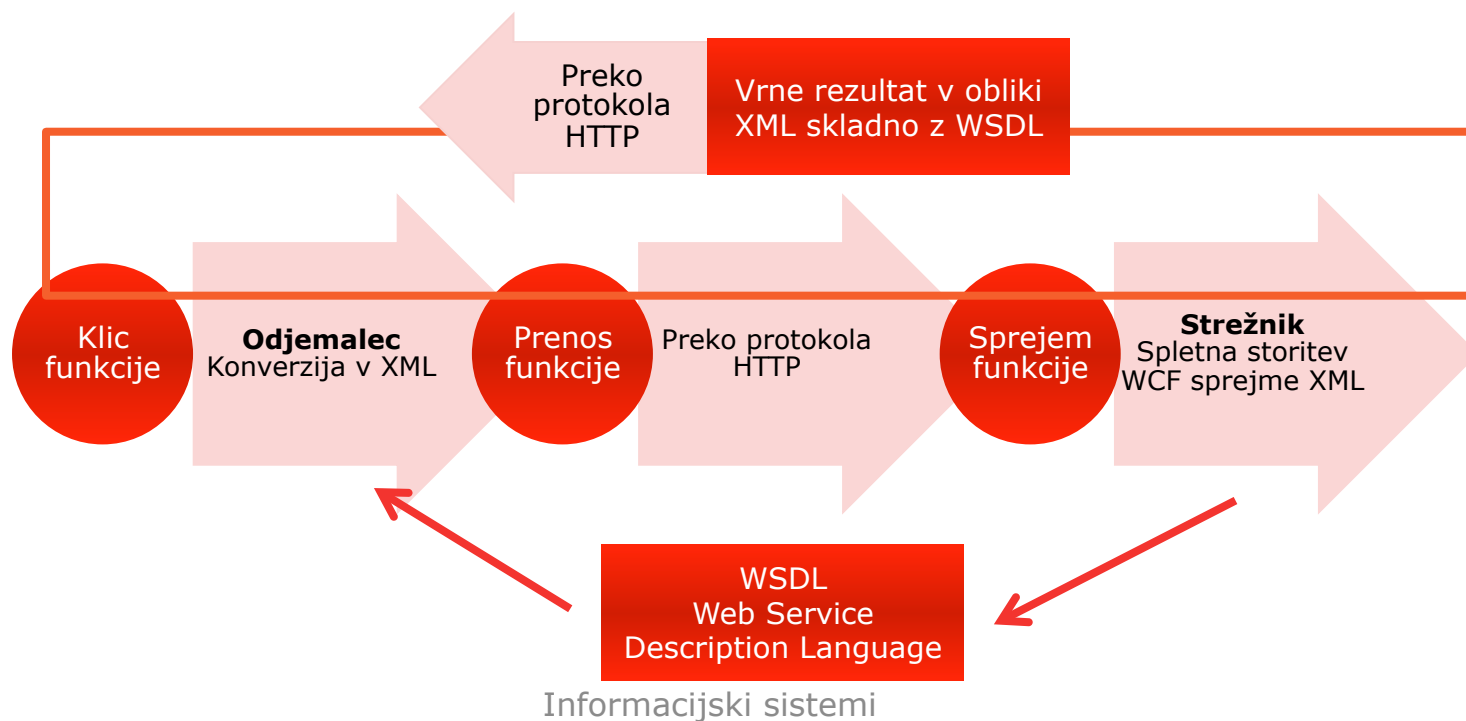


Web services

Spletne storitve

Spletne storitve ⁽¹⁾

- Windows Communication Foundation (WCF)
- Omogoča povezovanje sistemov ne glede na tehnologijo, platformo in jezik.



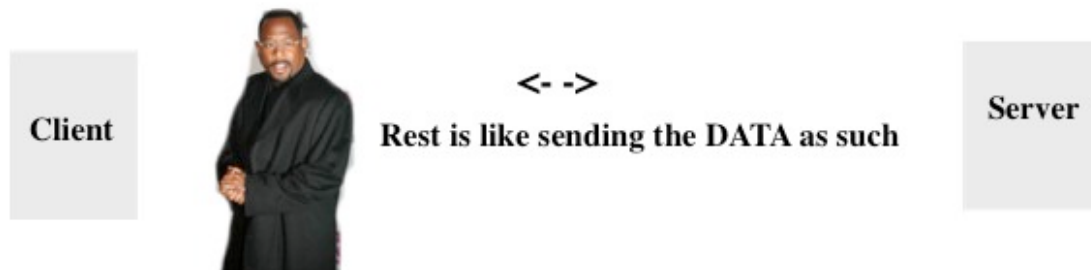
SOAP vs. REST

Consider "Martin Lawrence" as your data

SOAP

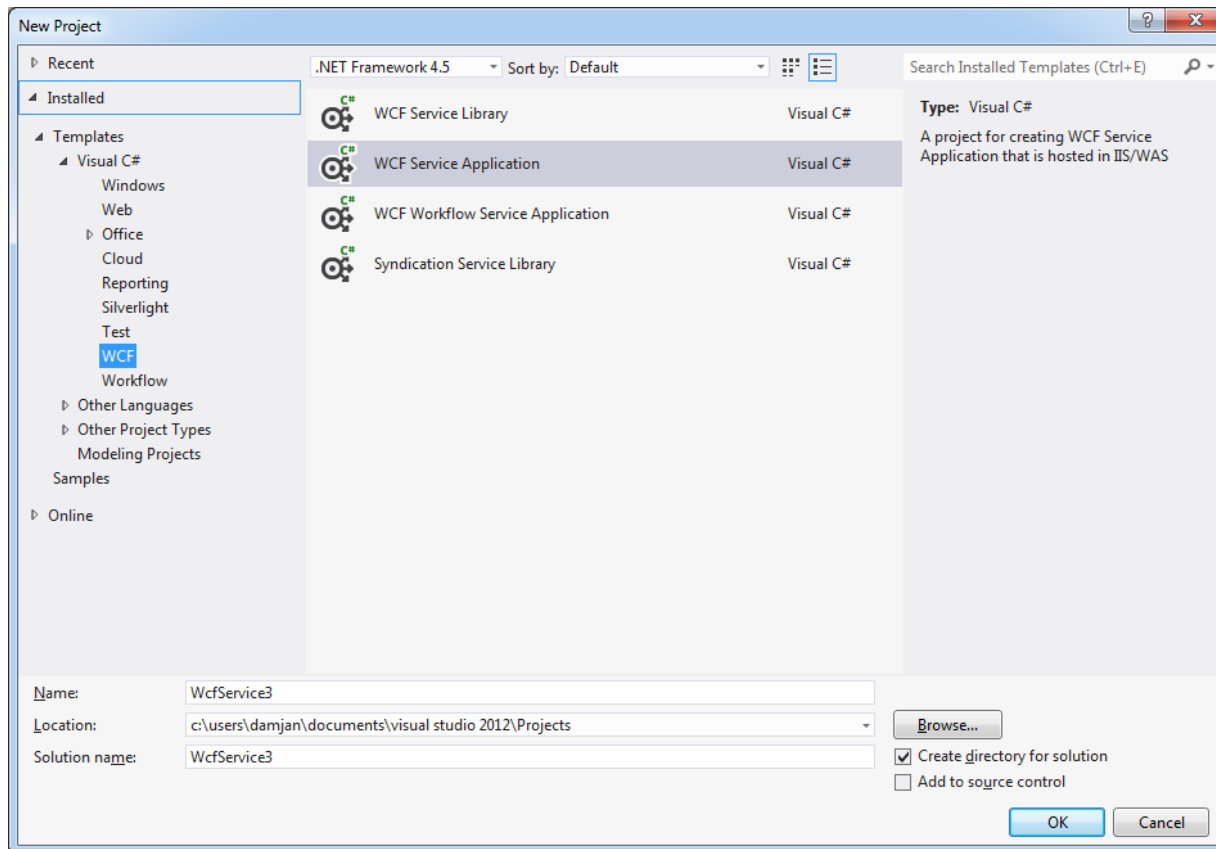


REST



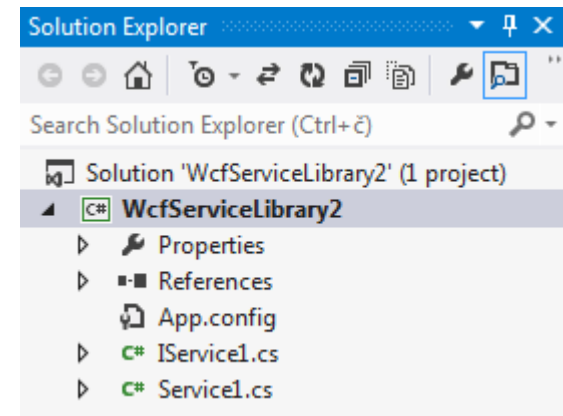
Spletne storitve⁽²⁾

Nova spletna storitev v okviru novega projekta (WCF Service Application Project)



Spletne storitve (3)

- Spletna storitev je definirana s pogodbo zapisano v obliki vmesnika (datoteka IService1.cs)
- Implementacija vmesnika je v pripadajočem razredu (datoteka Service1.svc)



Struktura vmesnika (1)

```
namespace WcfService
{
    [ServiceContract]
    public interface IService1
    {
        [OperationContract]
        string GetData(int value);

        [OperationContract]
        CompositeType GetDataUsingDataContract(CompositeType composite);
    }
    ...
}
```

Oznaka da gre za storitveno pogodbo (obvezno)

Ime storitve:
Spremenimo s spremembo imena datoteke .cs!

Oznaka da gre za pogodbo v zvezi z operacijo (obvezno)


Ime in signatura operacije, ki uporablja vgrajene tipe

Ime in signatura operacije, ki uporablja lastne oz. novo zgrajene tipe

Struktura vmesnika (2)

```
...  
[DataContract]  
public class CompositeType  
{  
    bool boolValue = true;  
    string stringValue = "Hello ";  
  
    [DataMember]  
    public bool BoolValue  
    {  
        get { return boolValue; }  
        set { boolValue = value; }  
    }  
  
    [DataMember]  
    public string StringValue  
    {  
        get { return stringValue; }  
        set { stringValue = value; }  
    }  
}  
}
```


Specifikacija lastnega
podatkovnega tipa



Struktura razreda

```
public class Service1 : IService1
{
    public string GetData(int value)
    {
        return string.Format("You entered: {0}", value);
    }

    public CompositeType GetDataUsingDataContract(CompositeType composite)
    {
        if (composite == null)
        {
            throw new ArgumentNullException("composite");
        }
        if (composite.BoolValue)
        {
            composite.StringValue += "Suffix";
        }
        return composite;
    }
}
```

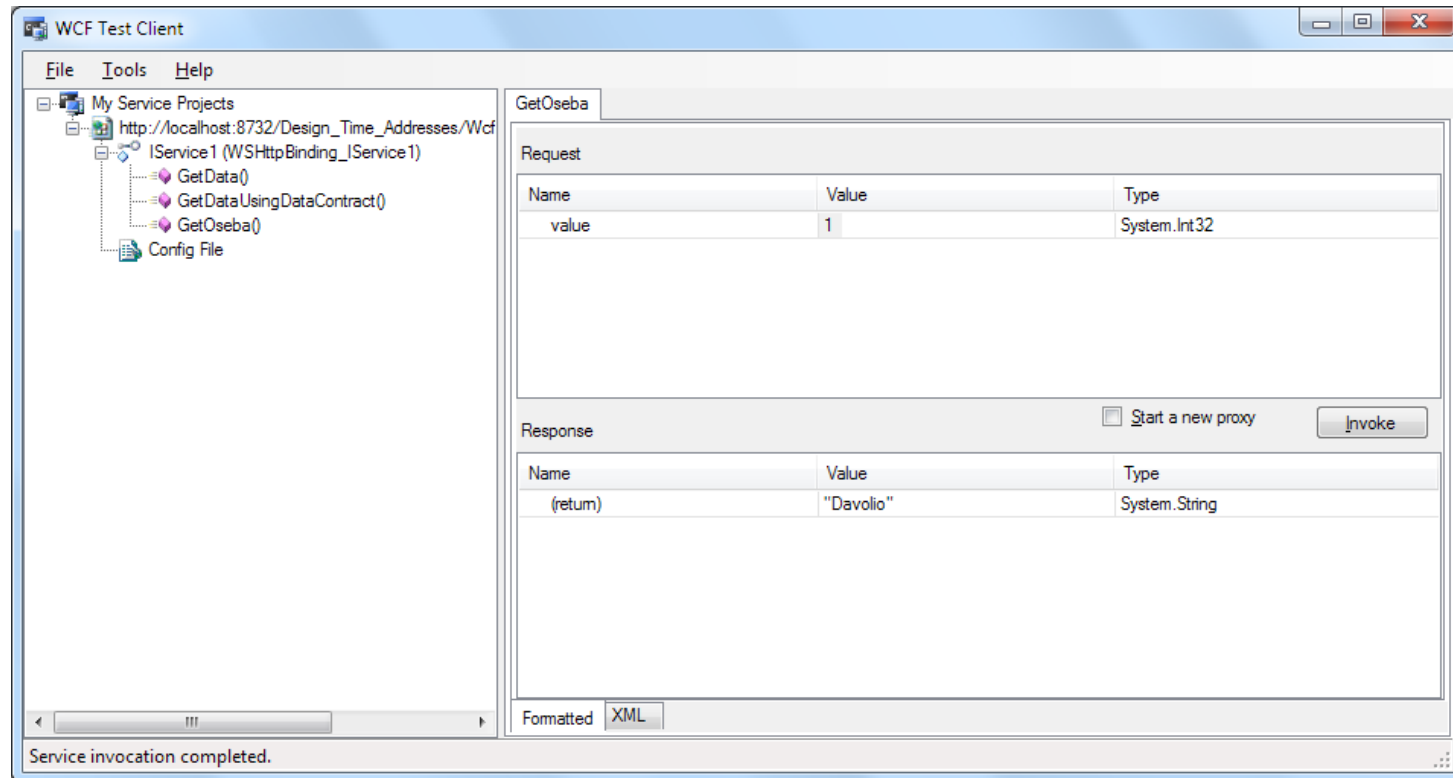


Razred, ki implementira storitveno pogodbo

Testiranje spletne storitve

WCF Test Client (WcfTestClient.exe)

C:\Program Files (x86)\Microsoft Visual Studio 12.0\Common7\IDE



Vaja 15

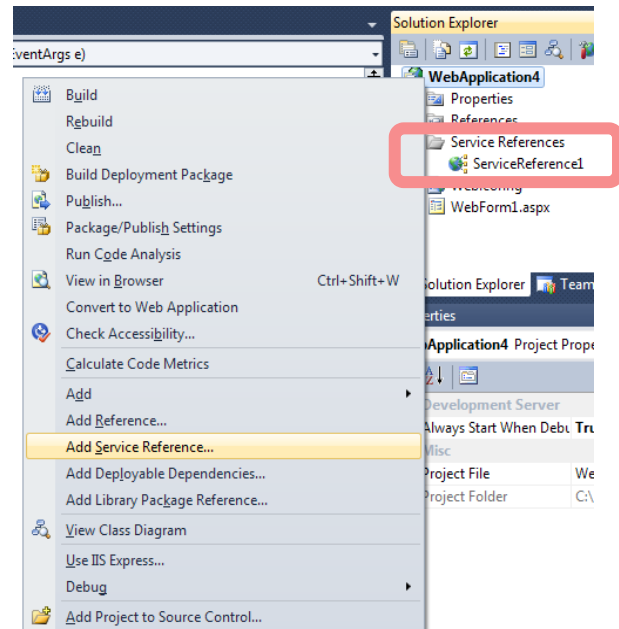
- Odprite nov projekt WCF Service Application
- Izdelajte spletno storitev, ki bo vsebovala operacijo VrniIme, ki bo vračala podatek o imenu za nek izbran ID v lokalnem seznamu imen.
- Preizkusite delovanje na lokalnem strežniku
- Preverjajte vnesene vrednosti in v primeru napačnega vnosa vrnite vrednost „Napaka! Ime ne obstaja.“

Dostop do zunanje spletne storitve (1)

- Dodamo referenco
- Poznati moramo naslov storitve
 - Če dodajamo lokalno storitev, lahko naslov najdemo v "WCF Test Client"

V kodi kjer bomo do storitve dostopali kreiramo nov objekt:

```
var websr = new ServiceReference1.Service1Client();
```



Dostop do zunanje spletne storitve (2)

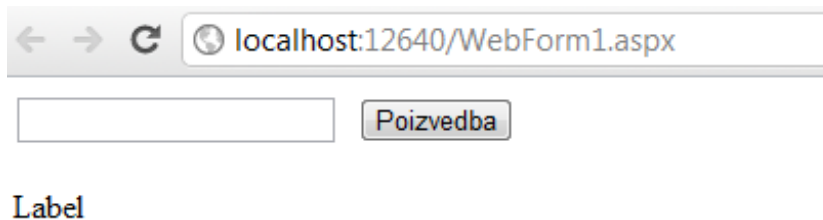
- Do operacij spletne storitve lahko dostopamo neposredno preko kreiranega objekta:

```
var websr = new ServiceReference1.Service1Client();  
  
Label1.Text=websr.GetOseba(1);
```

- Po uporabi spletne storitve, zaprite povezavo s klicem funkcije Close().

Vaja 16

- Poženite predhodno razvito spletno storitev na lokalnem strežniku in jo **pustite teči** (ne zapirajte VS).
- Odprite **ново instanco VS** in kreirajte Empty Web Application.
- Kreirajte novo referenco na lokalno spletno storitev.
- Izdelajte preprosto aplikacijo, ki bo ob pritisku na tipko prikazala rezultat, ki ga bo prebrala iz spletne storitve, ki ste jo izdelali v sklopu Vaje 15.



Polnjenje gradnika GridView iz spletne storitve (1)

- Gradnik *GridView* lahko iz spletne storitve napolnimo pomočjo gradnika ***ObjectDataSource***
- Postopek izdelave:

➤ Primer spletne storitve, ki jo bomo uporabili za prikaz delovanja

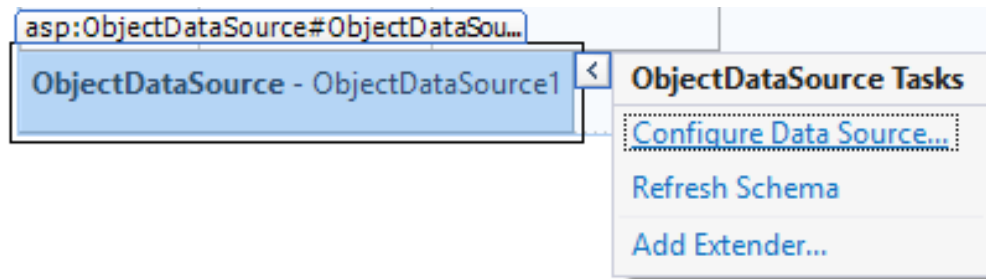
```
[ServiceContract]
public interface IService1
{
    [OperationContract]
    List<ProductEntity> GetProductList();
}

[DataContract]
public class ProductEntity
{
    [DataMember]
    public int ProductID { get; set; }
    [DataMember]
    public string ProductName { get; set; }
    [DataMember]
    public string ProductCategory { get; set; }
}
```

Polnjenje gradnika GridView iz spletne storitve (2)

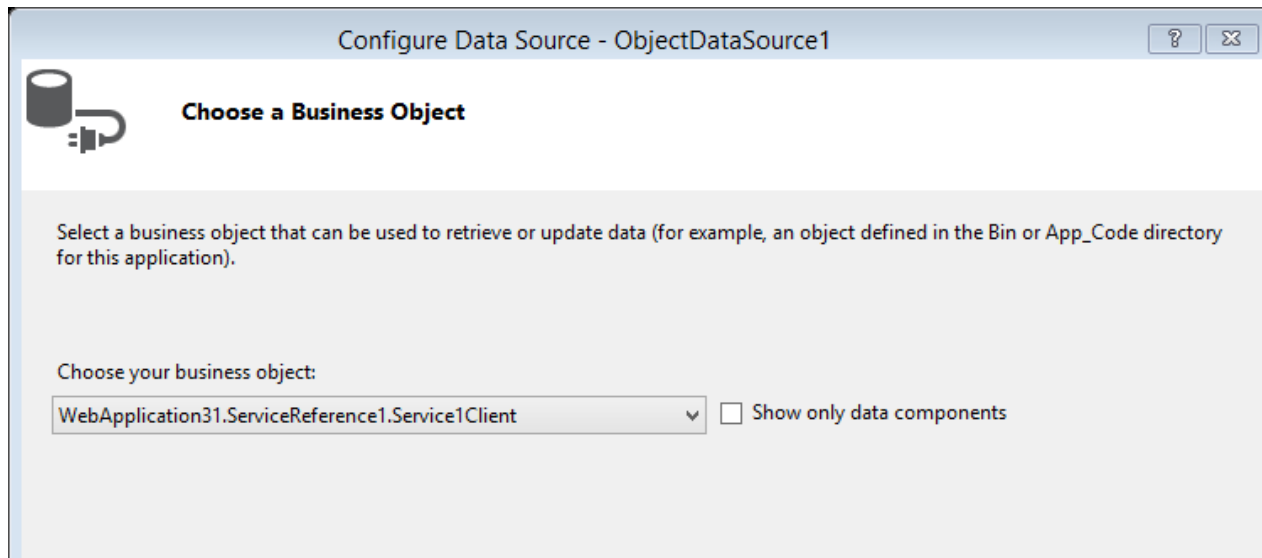
Dodamo referenco na spletno storitev

- **Desni klik na projekt → Add Service Reference...**
- Projekt rebuild-amo
- **Desni klik na projekt → Rebuild**
- Na spletno formo dodamo gradnik ObjectDataSource in ga nastavimo



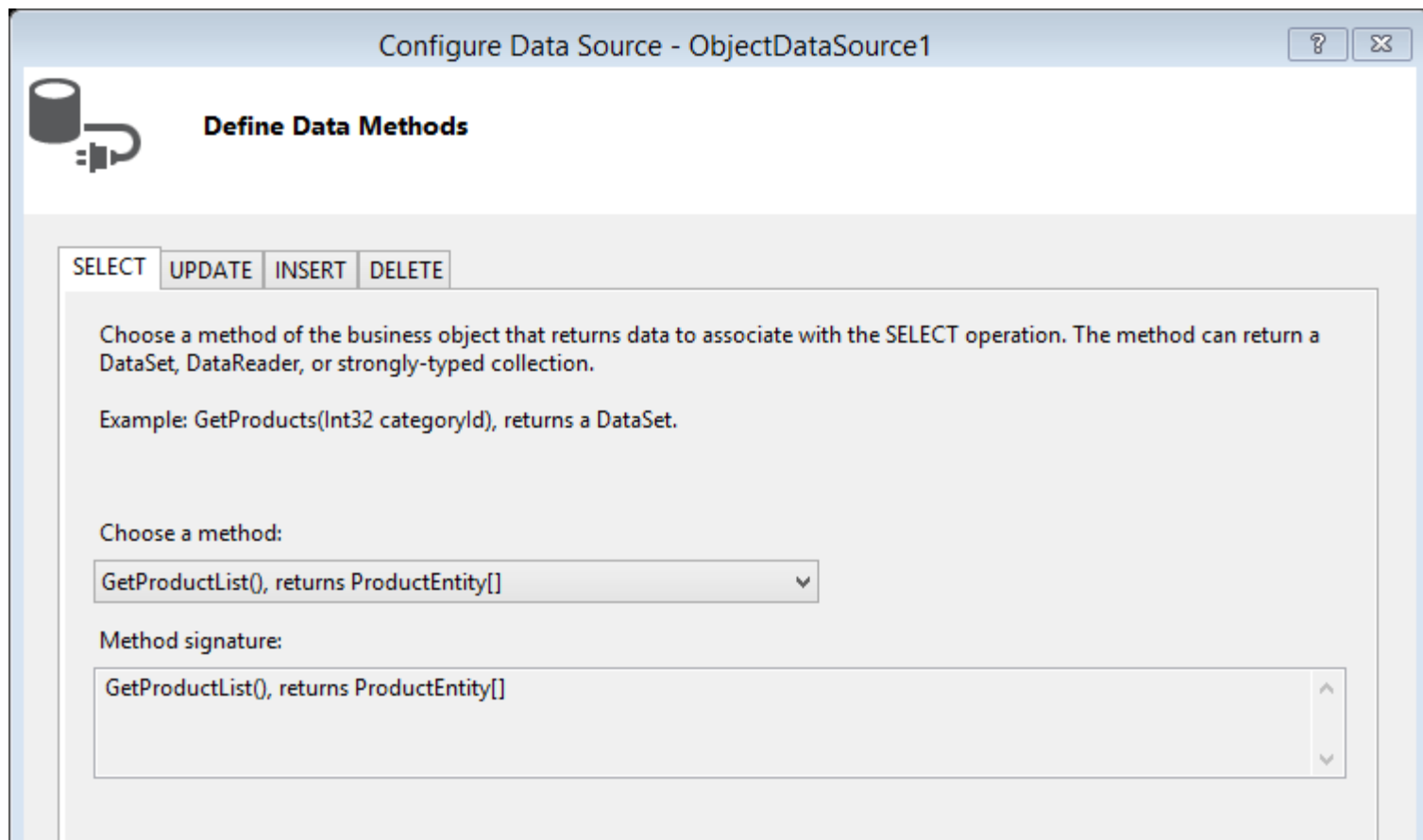
Polnjenje gradnika GridView iz spletne storitve (3)

Pod poljem *Choose your business object* izberemo postavko **NazivProjekt.NazivReferenceNaStoritev.NazivStoritveClient** in kliknemo *Next*



Polnjenje gradnika GridView iz spletne storitve (4)

Izberemo ustrezno operacijo storitve...



Polnjenje gradnika GridView iz spletne storitve (5)

➤ ObjectDataSource nastavimo kot DataSource pri gradniku GridView

Ostale nastavitve gradnika *GridView* so enake, kot pri uporabi *SqlDataSource*

The screenshot shows a Visual Studio IDE with a GridView control and its configuration tasks. The GridView is titled 'asp:GridView#GridView1' and displays a table with three columns: ProductID, ProductName, and ProductCategory. The data is as follows:

ProductID	ProductName	ProductCategory
0	abc	abc
1	abc	abc
2	abc	abc
3	abc	abc
4	abc	abc

Below the table, the data source is set to 'ObjectDataSource - ObjectDataSource1'. To the right, the 'GridView Tasks' pane is open, showing various configuration options:

- Auto Format...
- Choose Data Source: ObjectDataSource1 (dropdown)
- Configure Data Source...
- Refresh Schema
- Edit Columns...
- Add New Column...
- ☐ Enable Paging
- ☐ Enable Selection
- Add Extender...
- Edit Templates

Objavljanje spletne storitve

- Objavo spletne storitve (WCF Service Application) lahko izvedete enako, kot pri projektu tipa Web Application

Spletne storitve REST

- REST - Representational State Transfer
- CRUD: Create, Retrieve, Update, Delete
- HTTP Protokol
 - GET: R(Retrieve)
 - POST: U(Update)
 - PUT: C(Create)
 - DELETE: D(Delete)

REST storitev

- Protokol HTTP GET - dekorator "WebGet"
- Protokoli HTTP PUT, POST, DELETE – dekorator "WebInvoke"
- "UriTemplate"

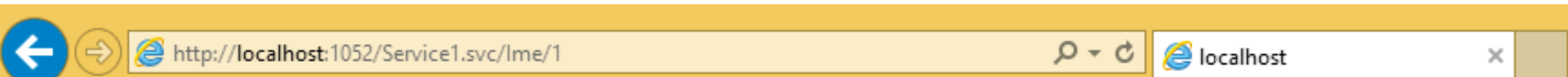
```
[ServiceContract]
1 reference
public interface IService1
{
    [OperationContract]
    [WebGet(UriTemplate = "Ime/{id}")]
    1 reference
    string VrniIme(string id);

    [OperationContract]
    [WebInvoke(UriTemplate = "DodajIme/{ime}/{priimek}")]
    1 reference
    void DodajIme(string ime, string priimek);
}
```

REST storitev – Web.config

```
<system.serviceModel>
  <services>
    <service name="WcfService1.Service1">
      <endpoint
        address=""
        behaviorConfiguration="restfulBehavior"
        binding="webHttpBinding"
        bindingConfiguration=""
        name="Service1"
        contract="WcfService1.IService1" />
      <host>
        <baseAddresses>
          <add baseAddress="http://localhost/service1"/>
        </baseAddresses>
      </host>
    </service>
  </services>
  <behaviors>
    <endpointBehaviors>
      <behavior name="restfulBehavior">
        <webHttp />
      </behavior>
    </endpointBehaviors>
  </behaviors>
```

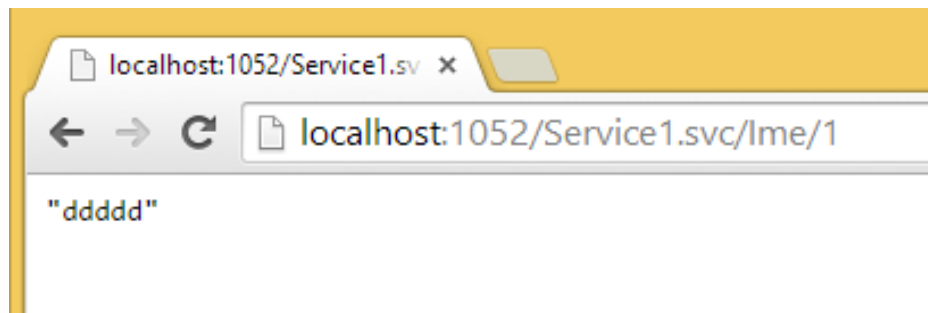
REST storitev – Primer



```
<?xml version="1.0"?>  
<string xmlns="http://schemas.microsoft.com/2003/10/Serialization/">dddd</string>
```

- Vrnjen rezultat privzeto v XML
- Sprememba v JSON:

```
[OperationContract]  
[WebGet(UriTemplate = "lme/{id}", ResponseFormat=WebMessageFormat.Json)]  
1 reference  
string VrniIme(string id);
```



REST storitev – Testiranje

- Postman REST Client (Google Chrome vtičnik)

The screenshot displays the Postman REST Client interface. At the top, the 'Normal' tab is selected, and the environment is set to 'No environment'. The URL bar contains 'http://localhost:1052/Service1.svc/DodajIme/Janez/Novak', and the HTTP method is 'POST'. Below the URL bar, the 'form-data' tab is active, showing a table with 'Key' and 'Value' columns. The 'Send' button is highlighted in blue. The response section shows a 'STATUS' of '200 OK' and a 'TIME' of '146271 ms'. The 'Body' tab is selected, and the 'Pretty' view is chosen. The response body contains a JSON object with a 'reference' field and a 'DodajIme' method call. A tooltip is visible over the 'celolme' field in the JSON, showing the value 'Janez Novak'.

Normal Basic Auth Digest Auth OAuth 1.0 No environment

http://localhost:1052/Service1.svc/DodajIme/Janez/Novak POST URL params Headers (0)

form-data x-www-form-urlencoded raw

Key Value Text

Send Preview Add to collection Reset

Body Headers (7) STATUS 200 OK TIME 146271 ms

Pretty Raw Preview JSON XML

```
1 reference
public void DodajIme(string ime, string priimek) {
    string celoIme = ime + " " + priimek;
    //dodaj v bazo celolme "Janez Novak"
}
```