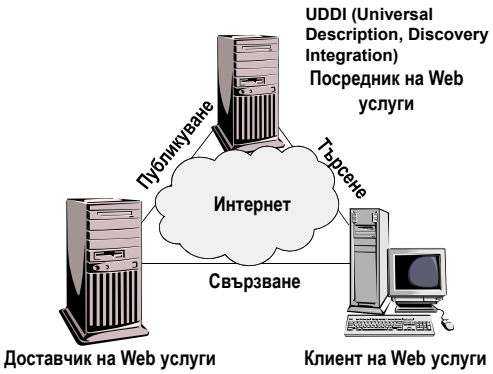
XML Web услуги



XML Web услуга е множество от функции, които:

- **се адресират чрез URL;**
- **достъпни са през Интернет;**
- **използват се като изграждащ блок за създаване на разпределени приложения.**

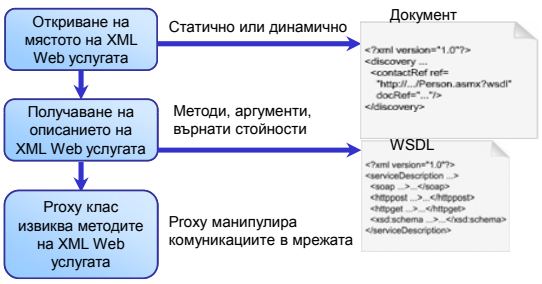
Използва

Интернет технологиите: HTTP, XML и SOAP

Основни елементи

- **доставчик на XML Web услуги;**
- **клиент на XML Web услуги;**
- **посредник на XML Web услуги.**

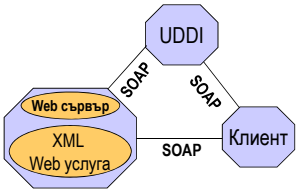
Принцип на действие на XML Web услуга



SOAP (Simple Object Access Protocol) е олекотен XML-базиран протокол за обмен на информация в децентрализирани, разпределени среди.

Характеристики

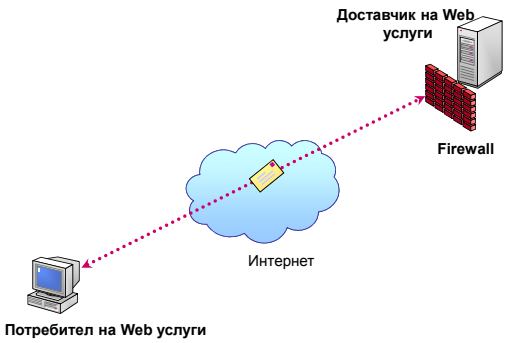
- **предава се по адрес;**
- **предава обекти, структури, множества от данни.**



WSDL (Web Services Description Language) – XML граматика за описание на Web услуга с езика на съобщенията, която приема и генерира WSDL документ.

WSDL документ – дефинира използваните типове в методите (операциите) на Web услугата и документите, които се обменят за всяка операция.

Създаване на Web услуги



1 **Създаване на Web услуга върху локалния Web сървър чрез шаблона ASP.NET Web Service**

<име на проект>

Клас <име на услуга>
– **наследник на** System.Web.Services.WebService**;**
– **файл** <име на услуга>.asmx**;**
– **файл с код** <име на услуга>.asmx.cs**.**

2 **Добавяне на атрибут** [WebService] **с име на пространството за Web услугата**

3 **Дефиниране на методи** <име на метод> **за XML Web услугата с атрибута** [WebMethod]

4 **Компилиране на XML Web услугата**

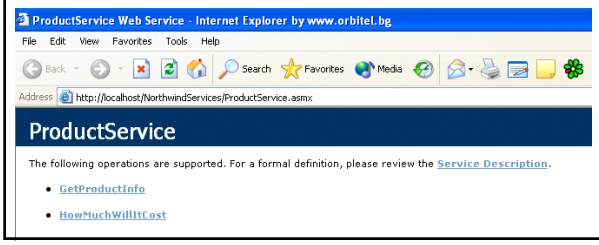
5 **Тестване на XML Web услугата**

InternetExplorer:

http://localhost/<име на проект>/<име на услуга>.asmx

Пример: Създаване на Web услуга с два Web метода:

- **Изчислява цената на определено количество от дадена стока от базата данни Northwind Traders.**
- **Връща данните за стока при зададено име на стока.**




```
using System;
using System.Collections;
using System.ComponentModel;
using System.Data;
using System.Diagnostics;
using System.Web;
using System.Web.Services;
using System.Data.SqlClient;

namespace NorthwindServices
{
  [WebService (Namespace = "http//pct.tu-sofia.bg/NorthwindServices")]

   public class ProductService : System.Web.Services.WebService
   {
     public ProductService()
     {
          InitializeComponent();
     }
```

име на сървър

Xml Web услуги

```
#region Component Designer generated code

private IContainer components = null;

private void InitializeComponent()
{
}

protected override void Dispose( bool disposing )
{
   if(disposing && components != null)
   {
        components.Dispose();
   }
   base.Dispose(disposing);
}

#endregion
```

```
[WebMethod]
public decimal HowMuchWillItCost (string productName, int howMany)
{ try
  { SqlConnection conn=new SqlConnection
      ("Persist Security Info=False;User ID=sa;Initial Catalog=Northwind;
       Data Source=localhost;Packet Size=4096;Workstation ID=localhost;
       Connect Timeout=300");
    SqlCommand cmd=new SqlCommand();
    cmd.CommandText =
      "SELECT UnitPrice FROM Products WHERE ProductName='" +
      productName+"'";
    cmd.Connection=conn;
    conn.Open();
    // ExecuteScalar връща първата колона от първия ред на заявката
    decimal price=(decimal)cmd.ExecuteScalar();
    conn.Close();
    return price*howMany;
  }
  catch(Exception e) {
    throw new Exception("Грешка при изчисляване на цената:" +
                        e.Message);   }
}
```

```
[WebMethod]
public Product GetProductInfo (string productName)
{
   Product product=new Product();
   try
   { SqlConnection conn=new SqlConnection
       ("Persist Security Info=False;User ID=sa;Initial Catalog=Northwind;
        Data Source=localhost;Packet Size=4096;Workstation ID=localhost;
       Connect Timeout=300");
     SqlCommand cmd=new SqlCommand();
     cmd.CommandText="SELECT * FROM Products WHERE
        ProductName='"+productName+"'";
     cmd.Connection=conn;
     conn.Open();
     SqlDataReader r=cmd.ExecuteReader();
     // Метод Read премества SqlDataReader към следващия запис;
     // връща true, ако има още записи
     if (r.Read())
     {
         // Методите GetInt32, GetInt16, GetString, GetDecimal, GetBoolean
         // връщат стойността на определената колона в съответния тип
          product.ProductID=r.GetInt32(0);
          product.ProductName=r.GetString(1);
          product.SupplierID=r.GetInt32(2);
          product.CategoryID=r.GetInt32(3);
          product.QuantityPerUnit=r.GetString(4);
          product.UnitPrice=r.GetDecimal(5);
          product.UnitsInStock=r.GetInt16(6);
          product.UnitsOnOrder=r.GetInt16(7);
          product.ReorderLevel=r.GetInt16(8);
          product.Discontinued=r.GetBoolean(9);
     }
     else
     { throw new Exception("Стоката " + productName + " не е намерена"); }
     r.Close();
     conn.Close();
     return product;
   }
   catch(Exception e)
    {  throw new Exception("Грешка при търсенето на стока: "+ e.Message);
    }
  }
}
```

```
// Project ⇒ Add Class…
using System;

namespace NorthwindServices
{
   public class Product
   {
         public int productID;
         private string productName;
         private int supplierID;
         private int categoryID;
         private string quantityPerUnit;
         private decimal unitPrice;
         private short unitsInStock;
         private short unitsOnOrder;
         private short reorderLevel;
         private bool discontinued;
         public int ProductID
         {
           get  {  return this.productID;  }
           set  {  this.productID=value;  }
         }
```

```
   public string ProductName
   {    get  {  return this.productName;  }
        set  {  this.productName=value;  }
   }
   public int SupplierID
   {    get  {  return this.supplierID;  }
        set  {  this.supplierID=value;  }
   }
   public int CategoryID
   {    get  {  return this.categoryID;  }
        set  {  this.categoryID=value;  }
   }
   public string QuantityPerUnit
   {    get  {  return this.quantityPerUnit;  }
        set  {  this.quantityPerUnit=value;  }
   }
   public decimal UnitPrice
   {    get  {  return this.unitPrice;  }
        set  {  this.unitPrice=value;  }
   }
```

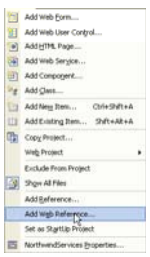
```
   public short UnitsInStock
   {    get  {  return this.unitsInStock;  }
        set  {  this.unitsInStock=value;  }
   }
   public short UnitsOnOrder
   {    get  {  return this.unitsOnOrder;  }
        set  {  this.unitsOnOrder=value;  }
   }
   public short ReorderLevel
   {    get  {  return this.reorderLevel;  }
        set  {  this.reorderLevel=value;  }
   }
   public bool Discontinued
   {    get  {  return this.discontinued;  }
        set  {  this.discontinued=value;  }
   }
   public Product()
   {
   }
  }
}
```

Създаване на клиент на Web услуги

1 **Създаване на клиентско приложение**

2 **Генериране на proxy**

Project ⇒ Add Web Reference



или

изпълнение на WSDL.exe

3 **Създаване на екземпляр на proxy**

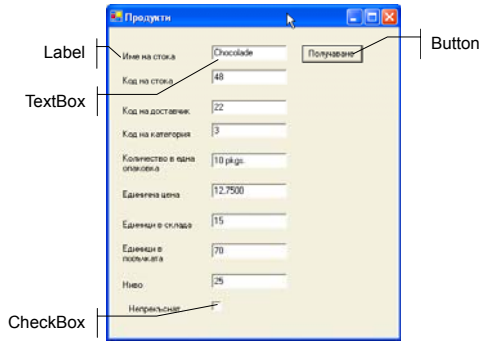
```
ProductService productService = new ProductService();
```

4 **Извикване на методите на XML Web услугите**

```
Product p =
        productService.GetProductInfo (productName.Text);
```

5 **Обработка на изключенията, прехвърляни от XML Web услугите**

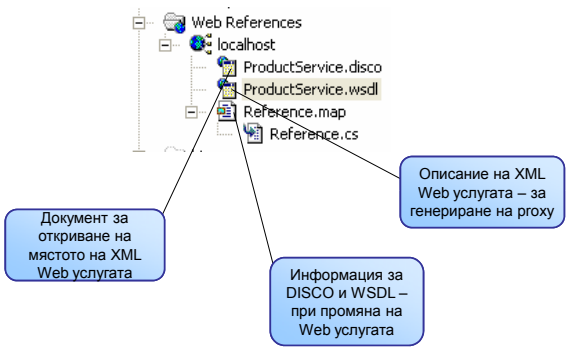
Пример: Реализиране на клиентско приложение, което консумира метода GetProductInfo на Web услугата ProductService.




```
using System;
using System.Drawing;
using System.Collections;
using System.ComponentModel;
using System.Windows.Forms;
using System.Data;
using ProductInfo.localhost;
namespace ProductInfo
{  public class Form1: System.Windows.Forms.Form
   {  private System.Windows.Forms.Label label1;
      private System.Windows.Forms.Label label2;
      private System.Windows.Forms.Label label3;
      private System.Windows.Forms.Label label4;
      private System.Windows.Forms.Label label5;
      private System.Windows.Forms.Label label6;
      private System.Windows.Forms.Label label7;
      private System.Windows.Forms.Label label8;
      private System.Windows.Forms.Label label9;
      private System.Windows.Forms.Label label10;
      private System.Windows.Forms.TextBox productName;
      private System.Windows.Forms.TextBox productID;
      private System.Windows.Forms.TextBox supplierID;
```

```
      private System.Windows.Forms.TextBox categoryID;
      private System.Windows.Forms.TextBox quantityPerUnit;
      private System.Windows.Forms.TextBox unitPrice;
      private System.Windows.Forms.TextBox unitsInStock;
      private System.Windows.Forms.TextBox unitsOnOrder;
      private System.Windows.Forms.TextBox reorderLevel;
      private System.Windows.Forms.Button getButton;
      private System.Windows.Forms.CheckBox discontinued;
      private System.ComponentModel.Container components = null;

      public Form1()
      {  InitializeComponent();  }

      protected override void Dispose( bool disposing ) {  …  }

      Windows Form Designer generated code

      [STAThread]
      static void Main()
      {
          Application.Run(new Form1());
      }
```

```
      private void getButton_Click (object sender, System.EventArgs e)
      {     // Създаване на екземпляр на proxy
            ProductService productService=new ProductService();
            try {  // Извикване на метод на XML Web услуга
               Product p = productService.GetProductInfo(productName.Text);
               productID.Text=p.ProductID.ToString();
               supplierID.Text=p.SupplierID.ToString();
               categoryID.Text=p.CategoryID.ToString();
               quantityPerUnit.Text=p.QuantityPerUnit;
               unitPrice.Text=p.UnitPrice.ToString();
               unitsInStock.Text=p.UnitsInStock.ToString();
               unitsOnOrder.Text=p.UnitsOnOrder.ToString();
               reorderLevel.Text=p.ReorderLevel.ToString();
               discontinued.Checked=p.Discontinued; }
            catch(Exception ex) { // Обработка на изключенията от Web услугата
               MessageBox.Show
                  ("Грешка при получаване на информация за стоката:" +
                  ex.Message,"Грешка",MessageBoxButtons.OK,
                  MessageBoxIcon.Error);     }
      }
   }
}
```



Файл ProductService.disco

```
<?xml version="1.0" encoding="utf-8"?>
<discovery xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns="http://schemas.xmlsoap.org/disco/">
  <contractRef
ref="http://localhost/NorthwindServices/ProductService.asmx?wsdl"
docRef="http://localhost/NorthwindServices/ProductService.asmx"
xmlns="http://schemas.xmlsoap.org/disco/scl/" />
  <soap address="http://localhost/NorthwindServices/ProductService.asmx"
xmlns:q1="http//contentmaster.com/NorthwindServices"
binding="q1:ProductServiceSoap"
xmlns="http://schemas.xmlsoap.org/disco/soap/" />
</discovery>
```

Файл ProductService.wsdl

```
<?xml version="1.0" encoding="utf-8"?>
<definitions xmlns:http="http://schemas.xmlsoap.org/wsdl/http/"
xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
xmlns:s="http://www.w3.org/2001/XMLSchema"
xmlns:s0="http//pct.tu-sofia.bg/NorthwindServices"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:tm="http://microsoft.com/wsdl/mime/textMatching/"
xmlns:mime="http://schemas.xmlsoap.org/wsdl/mime/"
targetNamespace="http//pct.tu-sofia.bg/NorthwindServices"
xmlns="http://schemas.xmlsoap.org/wsdl/">
  <types>
    <s:schema elementFormDefault="qualified"
    targetNamespace="http//pct.tu-sofia.bg/NorthwindServices">
      <s:element name="HowMuchWillItCost">
        <s:complexType>
          <s:sequence>
            <s:element minOccurs="0" maxOccurs="1" name="productName“
            type="s:string" />
```

```
          <s:element minOccurs="1" maxOccurs="1" name="howMany"
          type="s:int" />
        </s:sequence>
      </s:complexType>
    </s:element>
    <s:element name="HowMuchWillItCostResponse">
      <s:complexType>
        <s:sequence>
          <s:element minOccurs="1" maxOccurs="1"
          name="HowMuchWillItCostResult" type="s:decimal" />
        </s:sequence>
      </s:complexType>
    </s:element>
    <s:element name="GetProductInfo">
      <s:complexType>
        <s:sequence>
          <s:element minOccurs="0" maxOccurs="1" name="productName"
          type="s:string" />
        </s:sequence>
      </s:complexType>
    </s:element>
```

```
    <s:element name="GetProductInfoResponse">
      <s:complexType>
        <s:sequence>
          <s:element minOccurs="0" maxOccurs="1"
          name="GetProductInfoResult" type="s0:Product" />
        </s:sequence>
      </s:complexType>
    </s:element>
    <s:complexType name="Product">
      <s:sequence>
        <s:element minOccurs="1" maxOccurs="1" name="productID"
        type="s:int" />
        <s:element minOccurs="1" maxOccurs="1" name="ProductID"
        type="s:int" />
        <s:element minOccurs="0" maxOccurs="1" name="ProductName"
        type="s:string" />
        <s:element minOccurs="1" maxOccurs="1" name="SupplierID"
        type="s:int" />
        <s:element minOccurs="1" maxOccurs="1" name="CategoryID"
        type="s:int" />
```

```
        <s:element minOccurs="0" maxOccurs="1" name="QuantityPerUnit"
        type="s:string" />
        <s:element minOccurs="1" maxOccurs="1" name="UnitPrice"
        type="s:decimal" />
        <s:element minOccurs="1" maxOccurs="1" name="UnitsInStock"
        type="s:short" />
        <s:element minOccurs="1" maxOccurs="1" name="UnitsOnOrder"
        type="s:short" />
        <s:element minOccurs="1" maxOccurs="1" name="ReorderLevel"
        type="s:short" />
        <s:element minOccurs="1" maxOccurs="1" name="Discontinued"
        type="s:boolean" />
      </s:sequence>
    </s:complexType>
  </s:schema>
</types>
<message name="HowMuchWillItCostSoapIn">
  <part name="parameters" element="s0:HowMuchWillItCost" />
</message>
<message name="HowMuchWillItCostSoapOut">
  <part name="parameters" element="s0:HowMuchWillItCostResponse" />
```

```
</message>
<message name="HowMuchWillItCostSoapOut">
  <part name="parameters" element="s0:HowMuchWillItCostResponse"
/>
</message>
<message name="GetProductInfoSoapIn">
  <part name="parameters" element="s0:GetProductInfo" />
</message>
<message name="GetProductInfoSoapOut">
  <part name="parameters" element="s0:GetProductInfoResponse" />
</message>
<portType name="ProductServiceSoap">
  <operation name="HowMuchWillItCost">
    <input message="s0:HowMuchWillItCostSoapIn" />
    <output message="s0:HowMuchWillItCostSoapOut" />
  </operation>
  <operation name="GetProductInfo">
    <input message="s0:GetProductInfoSoapIn" />
    <output message="s0:GetProductInfoSoapOut" />
  </operation>
</portType>
```

```
  <binding name="ProductServiceSoap" type="s0:ProductServiceSoap">
   <soap:binding transport="http://schemas.xmlsoap.org/soap/http"
   style="document" />
   <operation name="HowMuchWillItCost">
    <soap:operation soapAction =
    "http//pct.tu-sofia.bg/NorthwindServices/HowMuchWillItCost"
    style="document" />
    <input>
     <soap:body use="literal" />
    </input>
    <output>
     <soap:body use="literal" />
    </output>
   </operation>
   <operation name="GetProductInfo">
    <soap:operation soapAction =
    "http//pct.tu-sofia.bg/NorthwindServices/GetProductInfo"
    style="document" />
    <input>
     <soap:body use="literal" />
    </input>
    <output>
```

```
      <soap:body use="literal" />
    </output>
   </operation>
  </binding>
  <service name="ProductService">
   <port name="ProductServiceSoap" binding="s0:ProductServiceSoap">
    <soap:address location =
    "http://localhost/NorthwindServices/ProductService.asmx" />
   </port>
  </service>
</definitions>
```

Файл Reference.map

```
<?xml version="1.0" encoding="utf-8"?>
<DiscoveryClientResultsFile xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
 <Results>
  <DiscoveryClientResult referenceType =
   "System.Web.Services.Discovery.ContractReference"
   url="http://localhost/NorthwindServices/ProductService.asmx?wsdl"
   filename="ProductService.wsdl" />
  <DiscoveryClientResult referenceType =
   "System.Web.Services.Discovery.DiscoveryDocumentReference"
   url="http://localhost/NorthwindServices/ProductService.asmx?disco"
   filename="ProductService.disco" />
 </Results>
</DiscoveryClientResultsFile>
```

Proxy клас – файл Reference.cs

```
namespace ProductInfo.localhost {
   using System.Diagnostics;
   using System.Xml.Serialization;
   using System;
   using System.Web.Services.Protocols;
   using System.ComponentModel;
   using System.Web.Services;

   [System.Diagnostics.DebuggerStepThroughAttribute()]
   [System.ComponentModel.DesignerCategoryAttribute("code")]
   [System.Web.Services.WebServiceBindingAttribute
     (Name="ProductServiceSoap",
      Namespace ="http//contentmaster.com/NorthwindServices")]

   public class ProductService :
     System.Web.Services.Protocols.SoapHttpClientProtocol {
      public ProductService() {
         this.Url = "http://localhost/NorthwindServices/ProductService.asmx";
      }
```

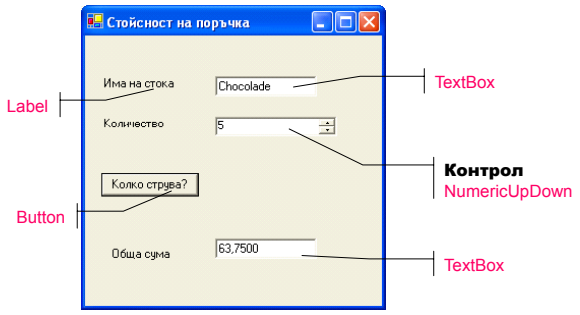
```
      [System.Web.Services.Protocols.SoapDocumentMethodAttribute
        ("http//pct.tu-sofia.bg/NorthwindServices/HowMuchWillItCost",
        RequestNamespace="http//pct.tu-sofia.bg/NorthwindServices",
        ResponseNamespace="http//pct.tu-sofia.bg/NorthwindServices",
        Use=System.Web.Services.Description.SoapBindingUse.Literal,
        ParameterStyle =
          System.Web.Services.Protocols.SoapParameterStyle.Wrapped)]
      public System.Decimal HowMuchWillItCost(string productName,
                                              int howMany) {
        object[] results = this.Invoke("HowMuchWillItCost", new object[] {
          productName,
          howMany});
         return ((System.Decimal)(results[0]));
      }

      public System.IAsyncResult BeginHowMuchWillItCost
        (string productName, int howMany,
         System.AsyncCallback callback, object asyncState) {
        return this.BeginInvoke("HowMuchWillItCost", new object[] {
          productName,
          howMany}, callback, asyncState);
      }
```

```
      public System.Decimal EndHowMuchWillItCost(System.IAsyncResult
        asyncResult) {
        object[] results = this.EndInvoke(asyncResult);
        return ((System.Decimal)(results[0]));
      }

      [System.Web.Services.Protocols.SoapDocumentMethodAttribute
        ("http//contentmaster.com/NorthwindServices/GetProductInfo",
        RequestNamespace="http//pct.tu-sofia.bg/NorthwindServices",
        ResponseNamespace="http//pct.tu-sofia.bg/NorthwindServices",
        Use=System.Web.Services.Description.SoapBindingUse.Literal,
        ParameterStyle =
          System.Web.Services.Protocols.SoapParameterStyle.Wrapped)]
      public Product GetProductInfo(string productName) {
        object[] results = this.Invoke("GetProductInfo", new object[] {
                productName});
        return ((Product)(results[0]));
      }
```

```
    public System.IAsyncResult BeginGetProductInfo(string productName,
        System.AsyncCallback callback, object asyncState) {
        return this.BeginInvoke("GetProductInfo", new object[] {
                    productName}, callback, asyncState);
    }

    public Product EndGetProductInfo(System.IAsyncResult asyncResult) {
        object[] results = this.EndInvoke(asyncResult);
        return ((Product)(results[0]));
    }
}

[System.Xml.Serialization.XmlTypeAttribute
  (Namespace="http//pct.tu-sofia.bg/NorthwindServices")]
public class Product {
  public int productID;
  public int ProductID;
  public string ProductName;
  public int SupplierID;
  public int CategoryID;
  public string QuantityPerUnit;
  public System.Decimal UnitPrice;
```

```
    public short UnitsInStock;
    public short UnitsOnOrder;
    public short ReorderLevel;
    public bool Discontinued;
  }
}
```

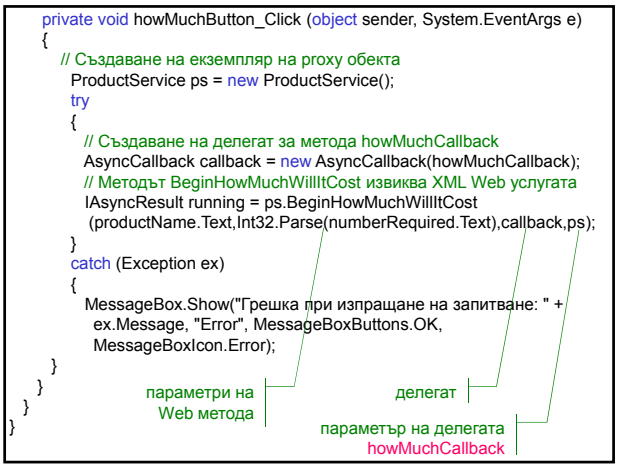
Пример: Реализиране на клиентско приложение, което консумира асинхронно метода HowMuchItWillCost **на Web услугата** ProductService**.**




```
using System;
using System.Drawing;
using System.Collections;
using System.ComponentModel;
using System.Windows.Forms;
using System.Data;
using OrderCost.localhost;

namespace OrderCost
{
  public class Form1 : System.Windows.Forms.Form
  { private System.Windows.Forms.Label label1;
    private System.Windows.Forms.Label label2;
    private System.Windows.Forms.Label label3;
    private System.Windows.Forms.Button howMuchButton;
    private System.Windows.Forms.TextBox productName;
    private System.Windows.Forms.TextBox totalCost;
    private System.Windows.Forms.NumericUpDown numberRequired;
    private System.ComponentModel.Container components = null;

    public Form1()
    {  InitializeComponent();  }

    protected override void Dispose( bool disposing ) { … }
```

```
Windows Form Designer generated code

[STAThread]
static void Main()
{
    Application.Run(new Form1());
}
```

```
  // Методът (callback) се изпълнява, когато методът HowMuchWillItCost
  // завърши; параметърът IAsyncResult съдържа информация за
  // данните, върнати от Web метода
  private void howMuchCallback(IAsyncResult ar)
  {
      try
      {
        // Свойството AsyncState съдържа обръщение към
        // proxy обекта, който изпълнява Web услуга
        ProductService ps=(ProductService)ar.AsyncState;
        // Методът EndHowMuchWillItCost връща резултата от
        // от Web метода
        decimal total=ps.EndHowMuchWillItCost(ar);
        totalCost.Text=total.ToString();
      }
      catch (Exception ex)
      {
        MessageBox.Show("Error Грешка при получаване на отговор: " +
          ex.Message, "Error", MessageBoxButtons.OK,
          MessageBoxIcon.Error);
    }
  }
```


private void howMuchButton_Click (object sender, System.EventArgs e)
{
// Създаване на екземпляр на proxy обекта
ProductService ps = new ProductService();
try
{
// Създаване на делегат за метода howMuchCallback
AsyncCallback callback = new AsyncCallback(howMuchCallback);
// Методът BeginHowMuchWillItCost извиква XML Web услугата
IAsyncResult running = ps.BeginHowMuchWillItCost
(productName.Text,Int32.Parse(numberRequired.Text),callback,ps);
}
catch (Exception ex)
{
MessageBox.Show("Грешка при изпращане на запитване: " +
ex.Message, "Error", MessageBoxButtons.OK,
MessageBoxIcon.Error);
}
}
}
}
параметри на Web метода
делегат
параметър на делегата howMuchCallback