

Praktische Einführung in *.NET*

Kai Stammerjohann

Inhalt

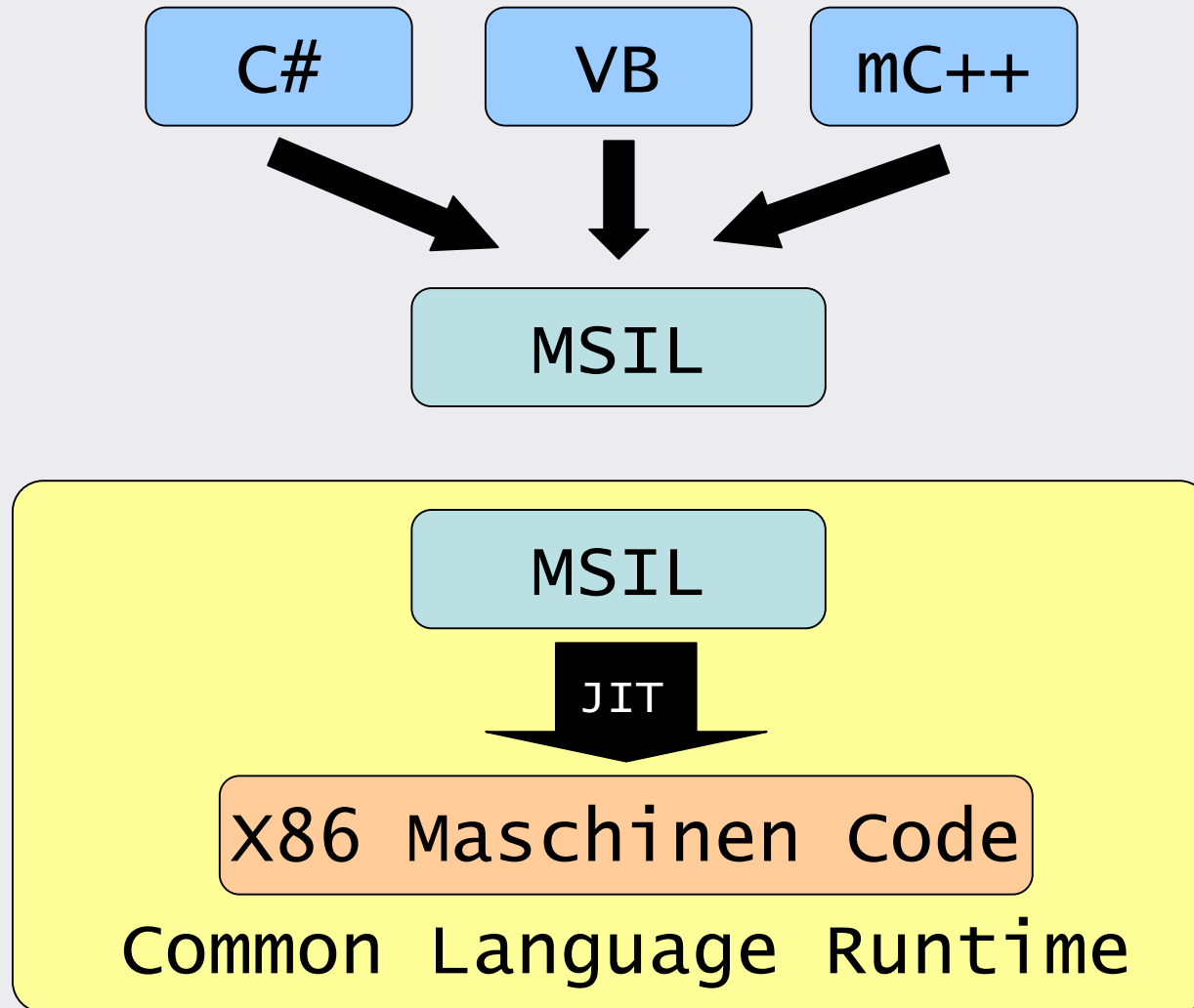
- CLR Eigenschaften
- Datenbankzugriff
- Web Service
- Webforms
- Schlussfolgerung

Inhalt

- CLR Eigenschaften
- Datenbankzugriff
- Web Service
- Webforms
- Schlussfolgerung

CLR Eigenschaften

Struktur



CLR Eigenschaften

Sprachübergreifende Programmierung

C#:

```
public class CSKlasse
{
    public void hallo(string text)
    {
        System.Console.WriteLine(text);
    }
}
```

mC++:

```
#using "CSKlasse.dll"

int main(void)
{
    CSKlasse *obj = new CSKlasse();

    obj->hallo("test");

    return 0;
}
```

CLR Eigenschaften

De-Compiler

Original:

```
public class CSKlasse
{
    public void hallo(string text)
    {
        System.Console.WriteLine(text);
    }
}
```

Zurückübersetzt:

```
C:\hs\Exemplar\release>exemplar CSKlasse.dll
```

```
public class CSKlasse {
    public void hallo(string text) {
        System.Console.WriteLine(text);
    }

    public CSKlasse() : base() {
    }
}
```

Inhalt

- CLR Eigenschaften
- **Datenbankzugriff**
- Web Service
- Webforms
- Schlussfolgerung

Datenbankzugriff

Herkömmlicher Datenbankzugriff

```
SqlConnection conn = new SqlConnection( ... );  
SqlCommand cmd = new SqlCommand("SELECT titel FROM hauptseminare", conn);  
  
conn.Open();  
  
SqlDataReader reader = cmd.ExecuteReader();  
  
while(reader.Read())  
{  
    Console.WriteLine("{0},{1}", reader.GetString(0), reader.GetString(1));  
}  
  
reader.Close();  
conn.Close();
```


Datenbankzugriff

Datenbankschema

The screenshot displays the Microsoft Visual Studio .NET IDE in design mode for a DataSet named 'EinschreiberDataSet.xsd'. The main design area shows three tables:

- Table: hauptseminare (hauptseminare)**

id	int
fachgebietid	int
titel	string
inhalt	string
betreuerid	int
eintragedatum	string
- Table: logins (logins)**

id	int
name	string
passwort	string
email	string
status	int
- Table: gruppen (gruppen)**

id	int
name	string

The 'Class View' on the right shows the project structure:

- test
 - EinschreiberDataSet
 - Test
 - Bases and Interfaces
 - Main(string[])

The 'Task List' at the bottom shows 'Build succeeded'.

Datenbankzugriff

ADO.NET Datenbankzugriff

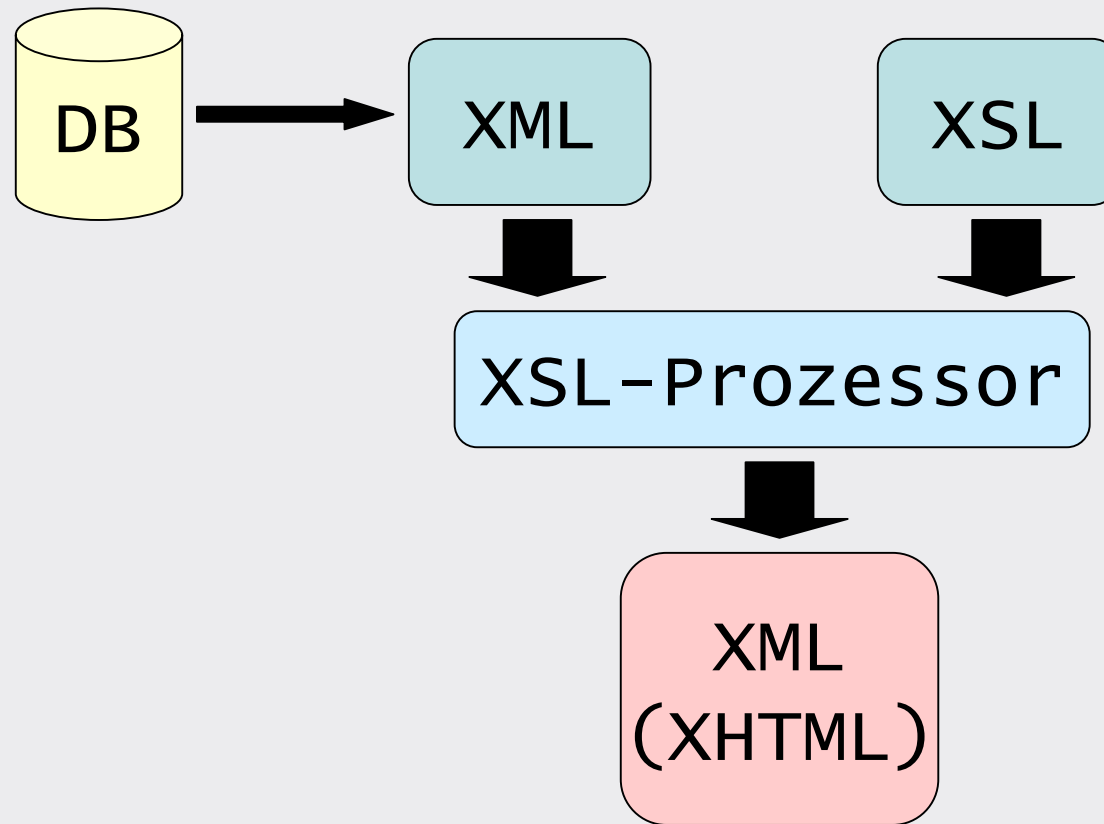
```
SqlConnection conn = new SqlConnection( ... );
SqlDataAdapter da = new SqlDataAdapter("SELECT * FROM hauptseminare", conn);
EinschreiberDataSet ds = new EinschreiberDataSet();

conn.Open();
da.Fill(ds, "hauptseminare");
conn.Close();

foreach(EinschreiberDataSet.hauptseminareRow row in ds.hauptseminare)
{
    Console.WriteLine("{0}, {1}", row.titel, row.inhalt);
}
```

Datenbankzugriff

XML/XSL



Datenbankzugriff

XML/XSL Datenbankzugriff

```
SqlConnection conn = new SqlConnection( ... );  
SqlDataAdapter da= new SqlDataAdapter("SELECT * FROM hauptseminare", conn);  
EinschreiberDataSet ds = new EinschreiberDataSet();  
  
conn.Open();  
da.Fill(ds, "hauptseminare");  
conn.Close();  
  
XmlDataDocument xml = new XmlDataDocument(ds);  
XslTransform xsl = new XslTransform();  
XmlTextWriter writer = new XmlTextWriter("output.html");  
  
xsl.Load("format.xsl");  
  
xsl.Transform(xml, null, writer);  
  
writer.close();
```

Inhalt

- CLR Eigenschaften
- Datenbankzugriff
- **web Service**
- webforms
- Schlussfolgerung

Definition

- Web Service = Dienst mit XML Schnittstelle
- Benutzung durch Programm oder menschlichen Anwender
- Kommunikation mit SOAP
- Registrierung in UDDI

Web Service

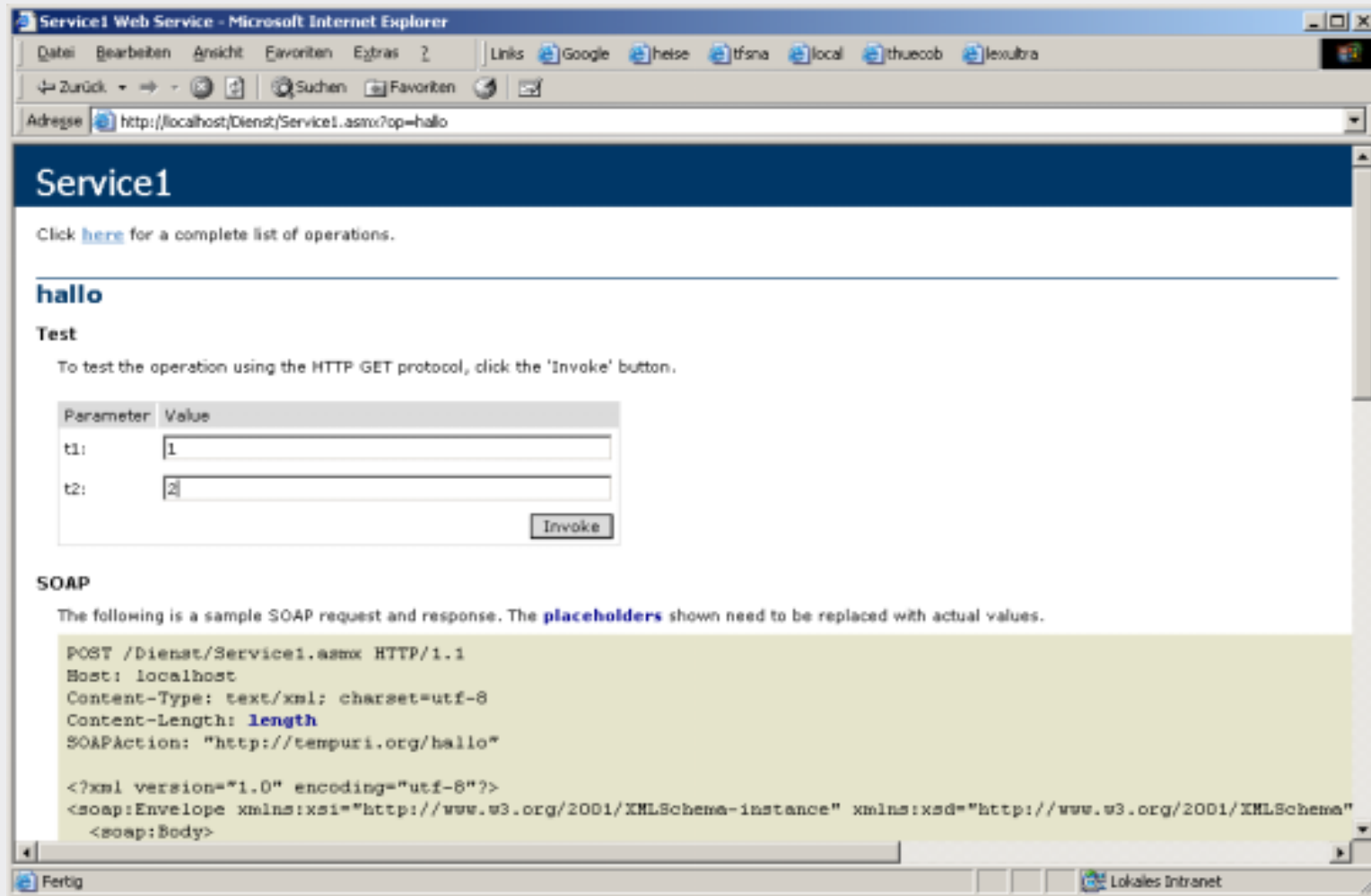
Dienst Server

```
public class Service1 : System.Web.Services.WebService
{
    [WebMethod]
    public Result execute(Param param)
    {
        return param.execute();
    }

    [WebMethod]
    public string hallo(string t1, string t2)
    {
        return "server meint zu " + t1 + " das da " + t2;
    }
}
```

Web Service

Dienst Anfrage



The screenshot shows a Microsoft Internet Explorer window titled "Service1 Web Service - Microsoft Internet Explorer". The address bar contains "http://localhost/Dienst/Service1.aspx?op=hallo". The page content includes a header "Service1", a link to a complete list of operations, a section titled "hallo" with a "Test" subsection, and a SOAP request and response example.

Service1

Click [here](#) for a complete list of operations.

hallo

Test

To test the operation using the HTTP GET protocol, click the 'Invoke' button.

Parameter	Value
t1:	<input type="text" value="1"/>
t2:	<input type="text" value="2"/>

SOAP

The following is a sample SOAP request and response. The **placeholders** shown need to be replaced with actual values.

```
POST /Dienst/Service1.aspx HTTP/1.1
Host: localhost
Content-Type: text/xml; charset=utf-8
Content-Length: length
SOAPAction: "http://tempuri.org/hallo"

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  <soap:Body>
```


Web Service

SOAP Anfrage Paket

```
<?xml version="1.0" encoding="utf-8"?>

<soap:Envelope
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">

  <soap:Body>

    <hallo xmlns="http://tempuri.org/">

      <t1>1</t1>
      <t2>2</t2>

    </hallo>

  </soap:Body>

</soap:Envelope>

entspricht: hallo("1", "2");
```

SOAP Antwort Paket

```
<?xml version="1.0" encoding="utf-8" ?>  
<string xmlns="http://uri/">server meint zu 1 das da 2</string>
```

Dienstbeschreibung

- WSDL (Web Service Description Language)
- Beschreibung aller Funktionen des Dienstes
- Beschreibung aller Parameter und Rückgaben
- Beschreibung aller Datentypen

Dienst Schnittstelle

```
public Result execute(Param param)
{
    return param.execute();
}
```

Dienst Datentypen

```
public class Param
{
    public Result execute()
    {
        return new Result("ergebnis");
    }
}

public class Result
{
    public string m;

    public Result(string m)
    {
        this.m = m;
    }
}
```

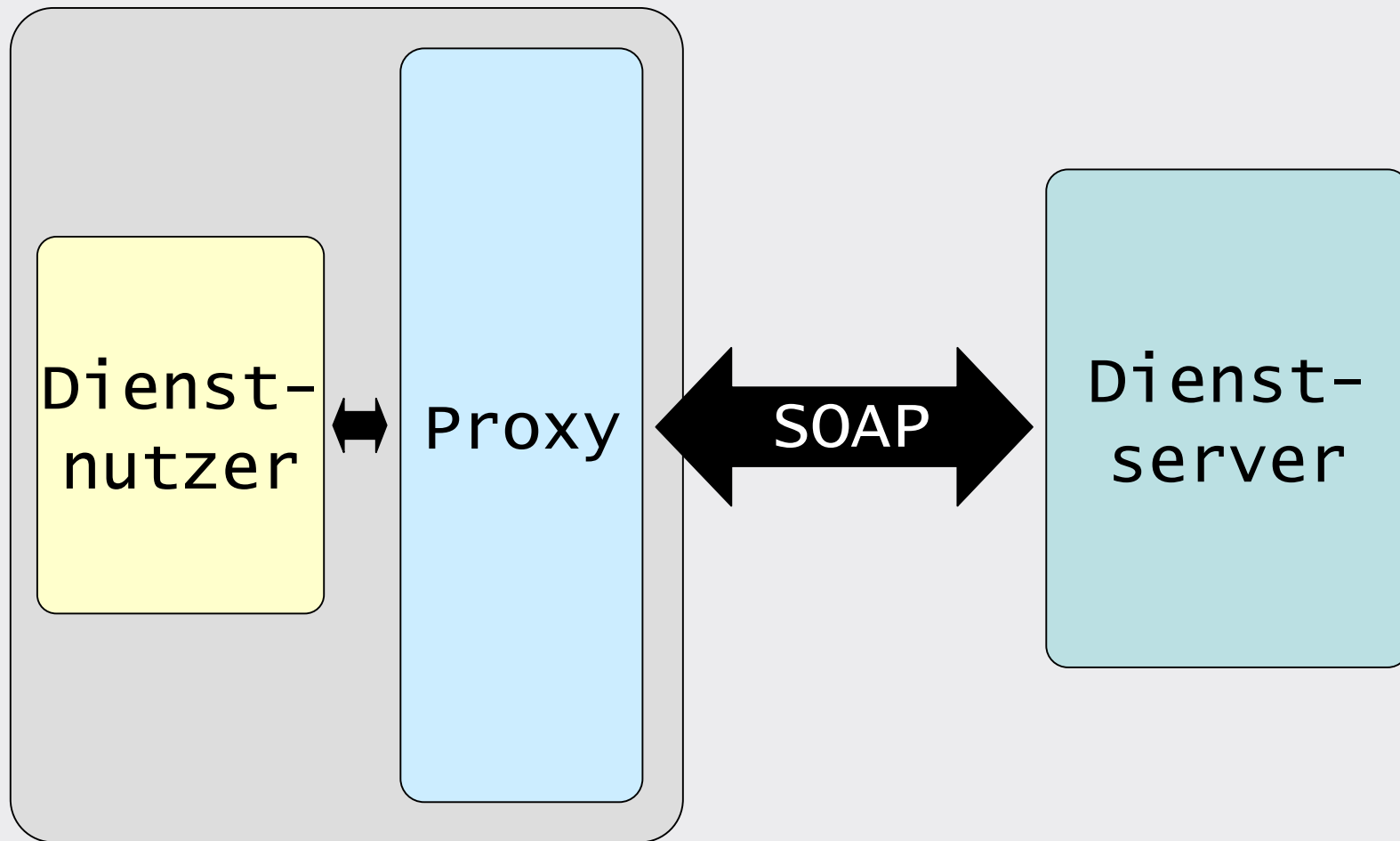
Web Service

WSDL

```
- <types>
- <s:schema elementFormDefault="qualified" targetNamespace="http://tempuri.org/">
  - <s:element name="execute">
    - <s:complexType>
      - <s:sequence>
        <s:element minOccurs="0" maxOccurs="1" name="param" type="s0:Param" />
      </s:sequence>
    </s:complexType>
  </s:element>
  <s:complexType name="Param" />
  - <s:element name="executeResponse">
    - <s:complexType>
      - <s:sequence>
        <s:element minOccurs="0" maxOccurs="1" name="executeResult" type="s0:Result" />
      </s:sequence>
    </s:complexType>
  </s:element>
  - <s:complexType name="Result">
    - <s:sequence>
      <s:element minOccurs="0" maxOccurs="1" name="m" type="s:string" />
    </s:sequence>
  </s:complexType>
  - <s:element name="hallo">
```

Web Service

Dienst Client - Proxy



Dienst Client

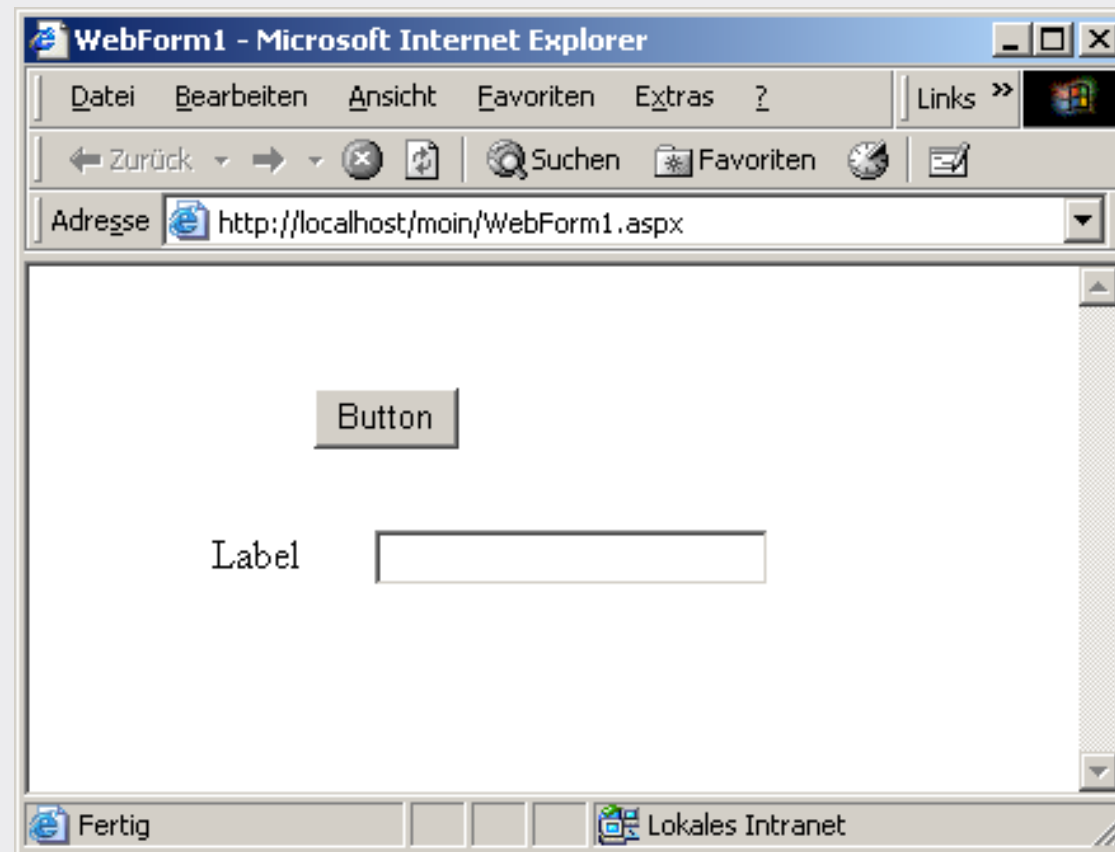
```
localhost.Service1 svc = new localhost.Service1();  
localhost.Result result = svc.execute(new localhost.Param());  
Console.WriteLine(result.m);  
Console.WriteLine(svc.hallo("1", "2"));
```


Inhalt

- CLR Eigenschaften
- Datenbankzugriff
- web Service
- **webforms**
- Schlussfolgerung

webforms

webform, webcontrol

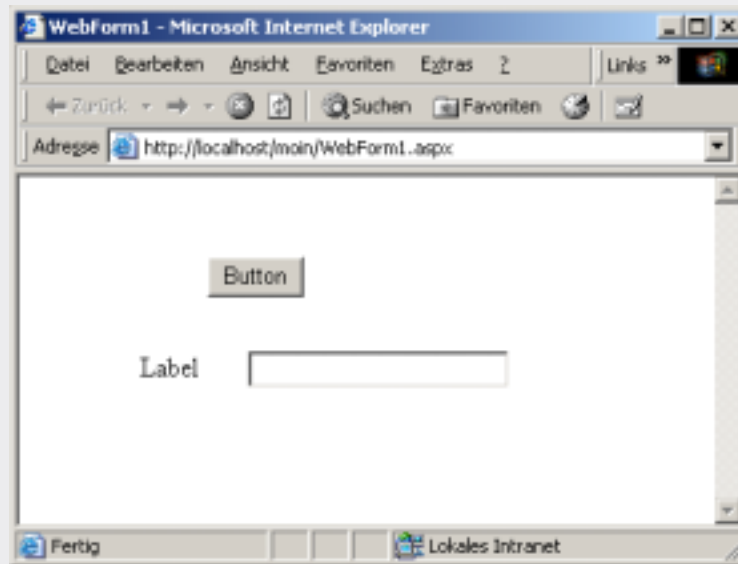


webform, webcontrol

- Zu jedem webcontrol gibt es eine zugehörige Klasse
Bsp: `<input>`: `System.Web.UI.WebControls.TextBox`
`<button>`: `System.Web.UI.WebControls.Button`
- Parallel zum (WYSIWYG) Entwurf des Webforms wird eine zugehörige webform Klasse erzeugt
- webformklasse enthält jedes enthaltene webcontrol als variable entsprechenden Typs

Webforms

webform, webcontrol Beispiel



```
public class WebForm1 : System.Web.UI.Page
{
    protected System.Web.UI.WebControls.TextBox TextBox1;
    protected System.Web.UI.WebControls.Button Button1;
    protected System.Web.UI.WebControls.Label Label1;

    . . .
}
```

Benutzerinteraktion

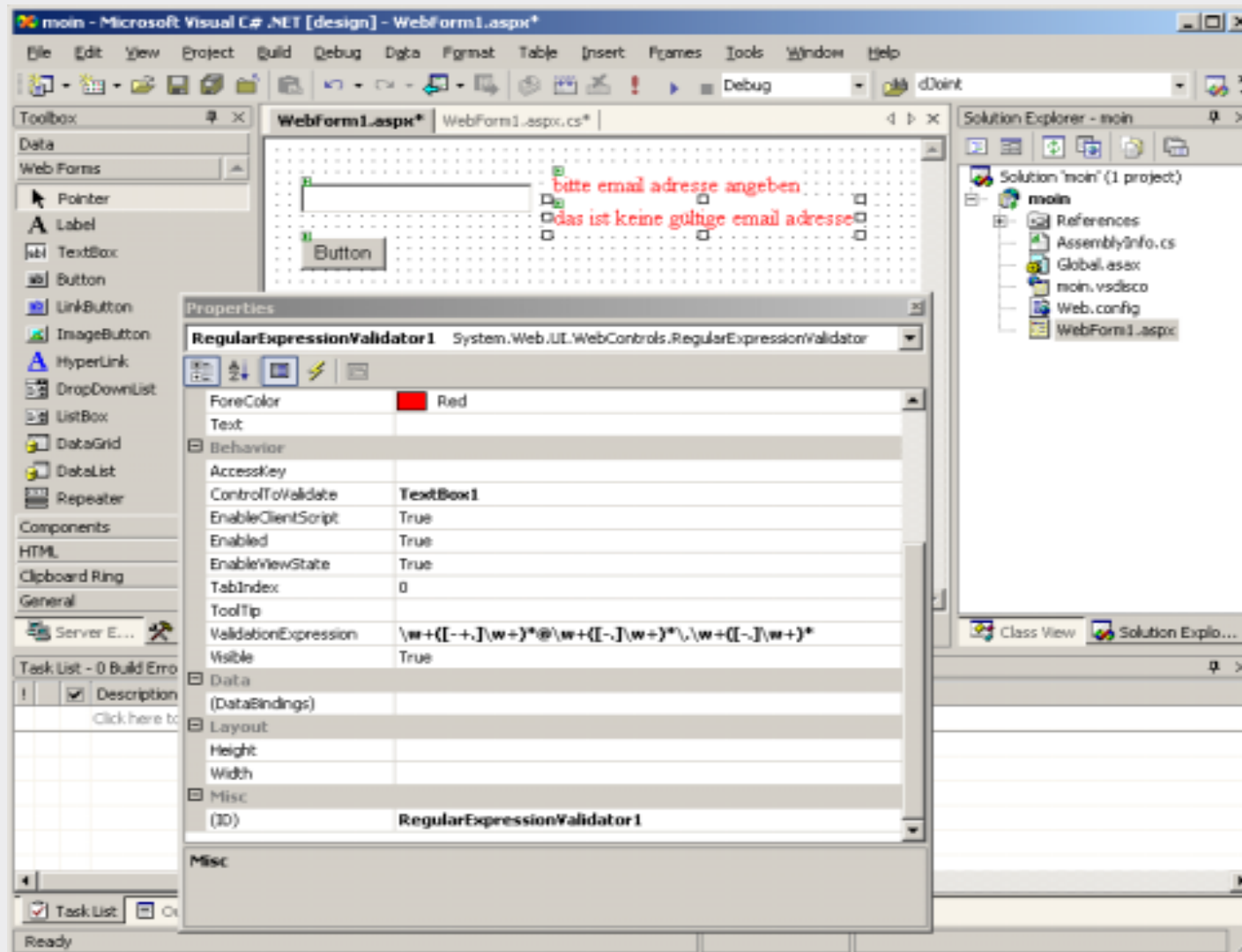
- verschiedene Eventhandler zu jedem webcontrol
- zB OnClick Handler für Buttons

Registrierung durch

```
this.Button1.Click += new System.EventHandler(this.Button1_Click);
```

webforms

validatoren



Inhalt

- CLR Eigenschaften
- Datenbankzugriff
- Web Service
- Webforms
- **Schlussfolgerung**

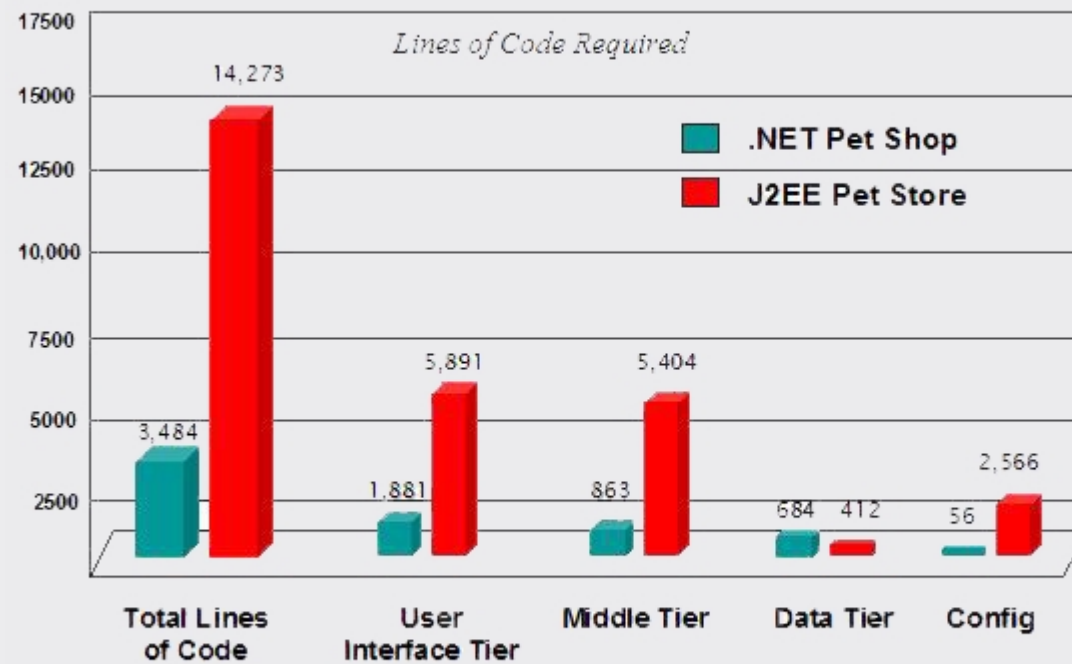
Schlussfolgerung

Alternative

- .NET Konkurrenzplattform ist J2EE
- J2EE Referenzinternetanwendung PetStore
- Nachimplementierung von VeriTest mit .NET

Schlussfolgerung

Implementing Sun's Java Petstore with Microsoft .NET



Ende

Fragen?