

WEB SERVICES & REMOTING

Nội dung

- Giới thiệu
- Tạo một Web service
- Gọi không đồng bộ đến Web services
- Cộng tác
- Hiệu suất
- An ninh
- Dịch vụ nâng cao
- .NET remoting

Giới thiệu

- .NET hỗ trợ mạnh và rất đầy đủ, những vấn đề phức tạp phía sau đã được che dấu, công việc lập trình trở nên đơn giản
- .NET hỗ trợ truyền thông với các ứng dụng phân tán

Tạo một Web Service

- Cần cài đặt IIS server
- Biến server REMOTE_HOST: địa chỉ IP để client kết nối
- Khởi tạo project mới, kiểu ASP.NET Web Service. Đường dẫn mặc định là IIS server (<http://localhost>)
- Thêm code sau vào file asmx:

Tạo một Web Service

```
[WebMethod]
public String[] getServerVariableNames()
{
    System.Collections.Specialized.NameValueCollection col;
    col=Context.Request.ServerVariables;
    String[] arr = col.AllKeys;
    return arr;
}
```

Tạo một Web Service

- Chú ý thuộc tính [WebMethod] phải đặt trước các tên hàm
- Mảng trả về từ hàm này sẽ được ghi nhận, nó sẽ chứa các chuỗi như REMOTE_HOST, HTTP_USER_AGENT
- Để lấy được giá trị của các biến chúng ta phải hiện thực thêm một hàm sau:

Tạo một Web Service

```
[WebMethod]
public string[] getServerVariable(string
variableName)
{
    System.Collections.Specialized.NameValueCollection col;
    col=Context.Request.ServerVariables;
    String[] arr = col.GetValues(variableName);
    return arr;
}
```

Tạo một Web Service

- Hàm trên trả về giá trị của một biến server
HTTP_ACCEPT liệt kê các kiểu MIME là trình duyệt có thể hiển thị
- Thực hiện chương trình trên, trình duyệt sẽ mở ra với **nội dung XML được định dạng với SOAP**
- Chú ý biến server như REMOTE_ADDR

Dùng Web Service

- Tạo project mới kiểu Windows Form, chọn Project→Add Web Reference và nhập vào URL của file ASMX được tạo trong ví dụ trước
- Tạo list view trên form, tên lvServerVariables, 1 button tên btnPopulate
- Code xử lý một số sự kiện:

Dùng Web Service

```
private void Form1_Load(object sender,  
System.EventArgs e)  
{  
    lvServerVariables.View=View.Details;  
    lvServerVariables.Columns.Add("Name",  
lvServerVariables.Width/2,  
HorizontalAlignment.Left);  
    lvServerVariables.Columns.Add("Value",  
lvServerVariables.Width/2,  
HorizontalAlignment.Left);  
}
```

Dùng Web Service

```
private void btnPopulate_Click(object sender,
System.EventArgs e)
{
    string[] serverVariableNames;
    localhost.Service1 webservice = new
localhost.Service1();
    serverVariableNames =
webservice.getServerVariableNames();
    lvServerVariables.Items.Clear();
    foreach (string serverVariableName in
serverVariableNames)
{
```

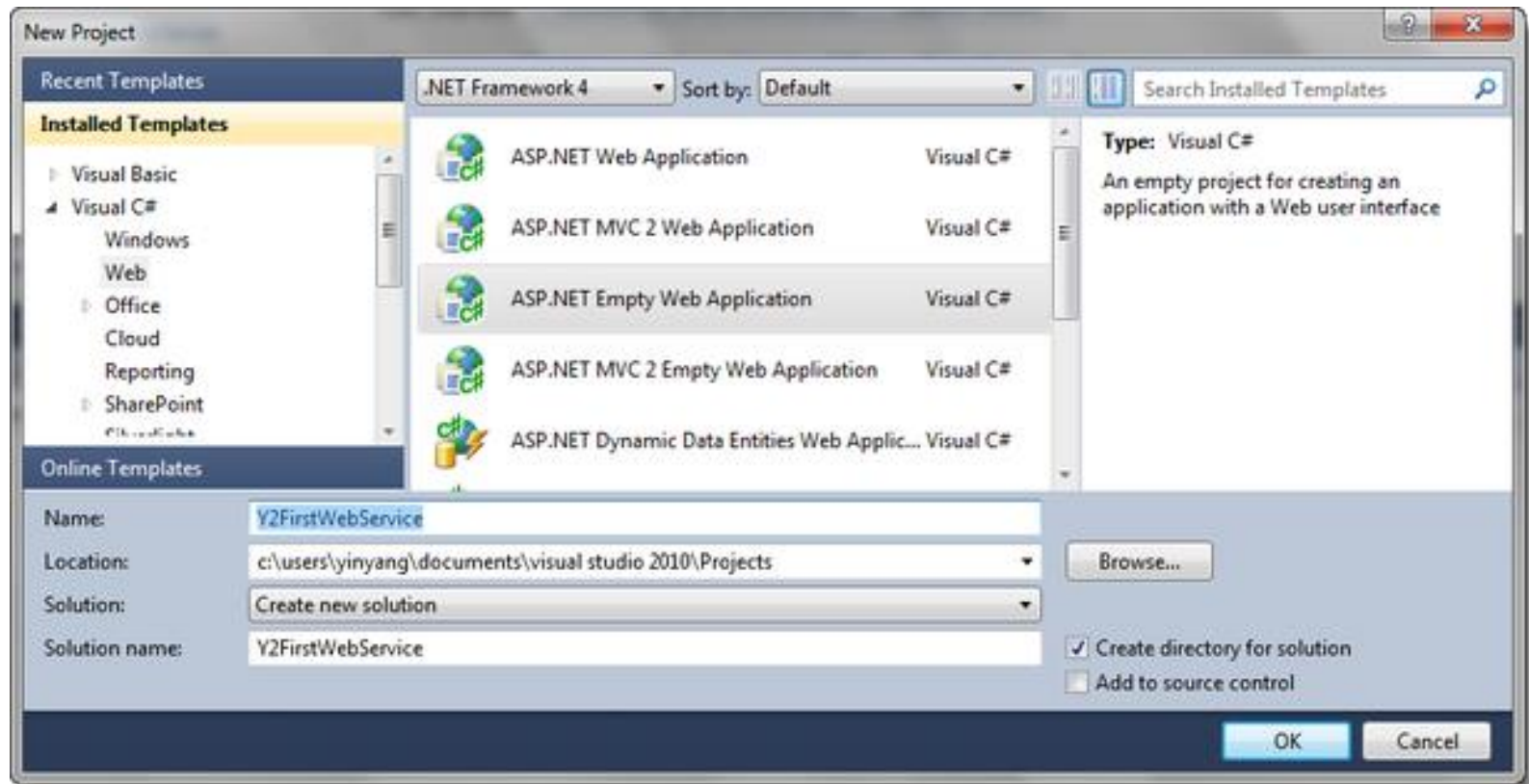
Dùng Web Service

```
ListViewItem lvItem = new ListViewItem();
lvItem.Text = serverVariableName;
string[] serverVariableValues;
serverVariableValues =
webservice.getServerVariable(serverVariableName);
if (serverVariableValues!=null)
    lvItem.SubItems.Add(serverVariableValues[0]);
lvServerVariables.Items.Add((ListViewItem)lvItem.Clone());
}
}
```

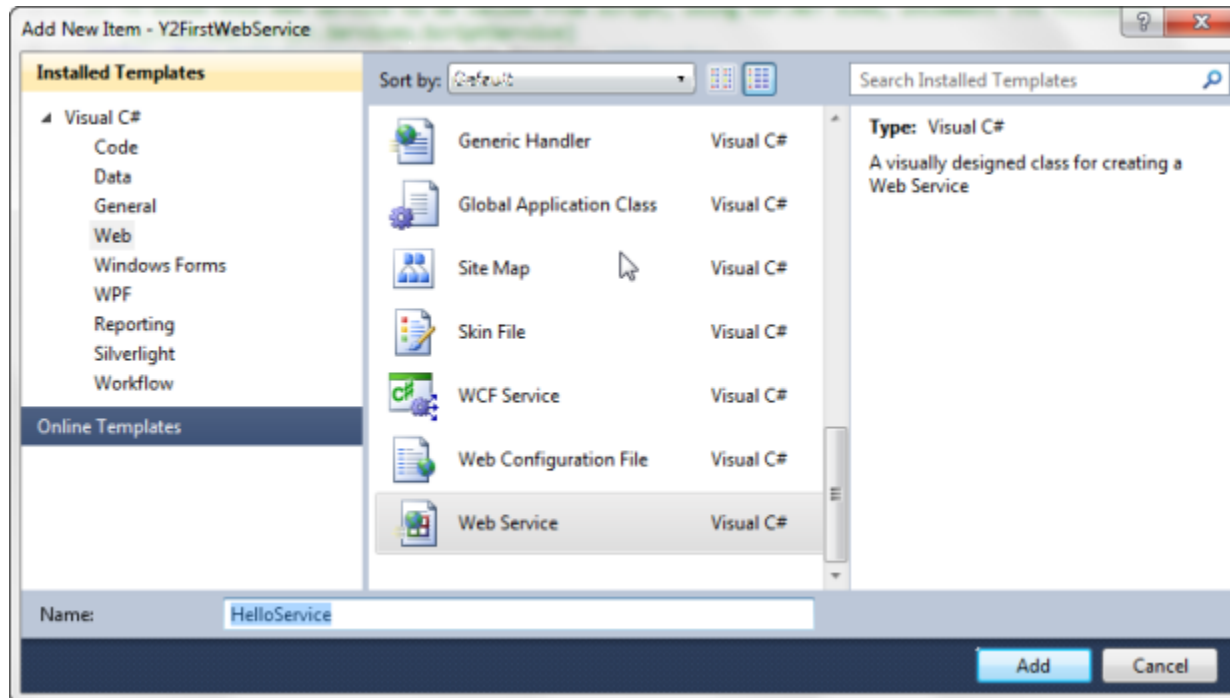
Dùng Web Service



Tạo Web Service Proxy



Tạo Web Service Proxy



Tạo Web Service Proxy

- Trước tiên cần tạo một Web Service Server (lưu trữ với tên MathService.asmx trong thư mục C:\inetpub\wwwroot\test) như sau:

```
<%@ WebService Language="c#"
Class="MathService"%>
using System;
using System.Web.Services;
[WebService(Namespace="http://localhost/test")]
public class MathService : WebService
{
```


Tạo Web Service Proxy

```
[WebMethod]
public int Add(int a, int b)
{
    int answer;
    answer = a + b;
    return answer;
}

[WebMethod]
public int Subtract(int a, int b)
{
    int answer;
    answer = a - b;
    return answer;
}
```

Tạo Web Service Proxy

```
[WebMethod]
public int Multiply(int a, int b)
{
    int answer;
    answer = a * b;
    return answer;
}
```

```
[WebMethod]
public int Divide(int a, int b)
{
```

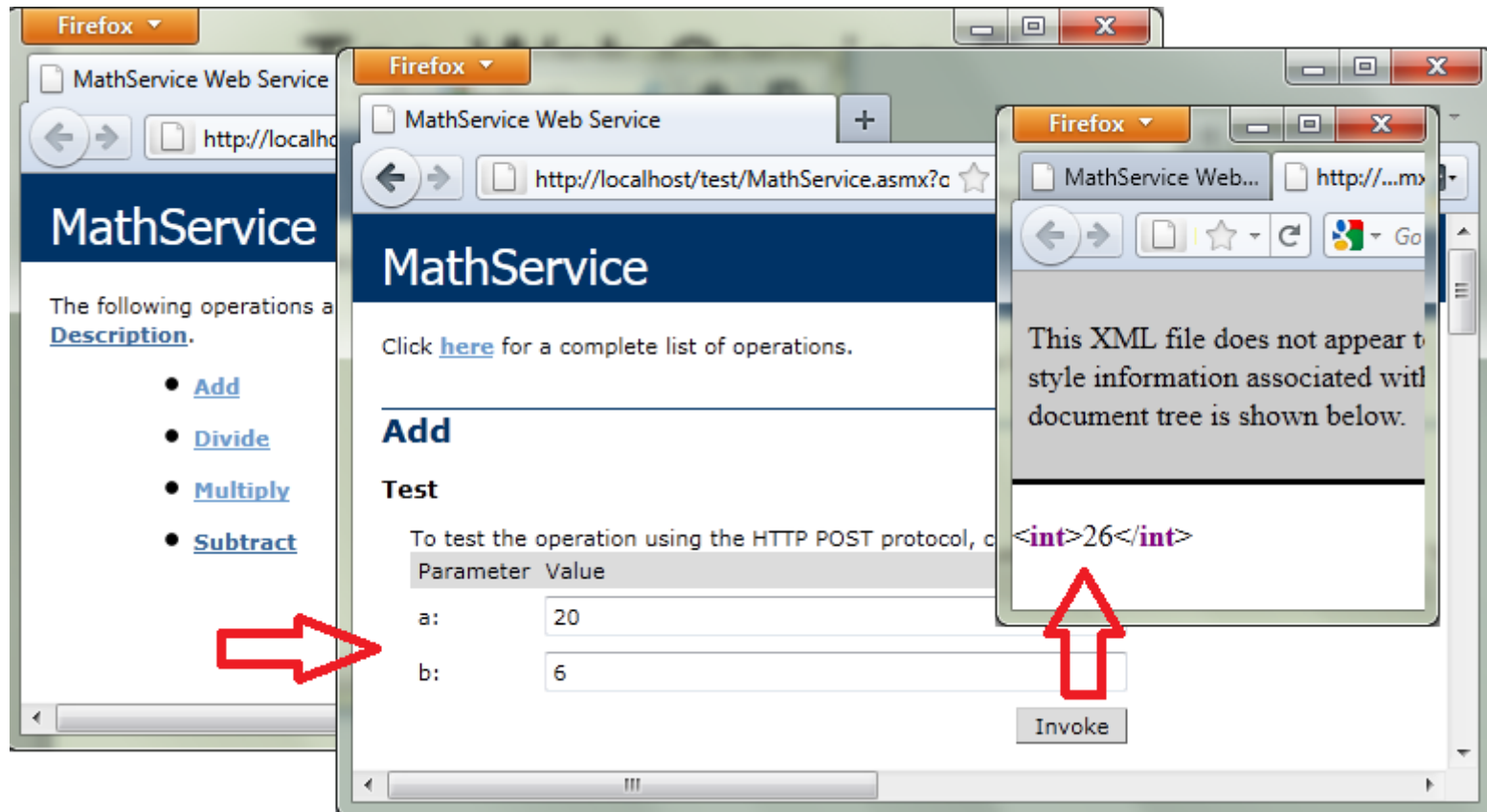
Tạo Web Service Proxy

```
int answer;  
if (b != 0)  
{  
    answer = a / b;  
    return answer;  
} else  
    return 0;  
}  
}
```

Tạo Web Service Proxy

- Đến đây thì chúng ta có thể thử nghiệm xem Web Service trên hoạt động như thế nào
- Sử dụng trình duyệt, nhập URI:
<http://localhost/test/MathService.asmx>
- Kết quả như hình minh họa

Tạo Web Service Proxy



Tạo Web Service Proxy

- Tạo một Web Service Proxy từ ứng dụng WSDL (có thể tìm thấy tại C:\Program Files (x86)\Microsoft SDKs\Windows\v7.0A\Bin)
- Thực hiện công việc trên qua dòng lệnh:
wsdl <http://localhost/test/MathService.asmx>
- Sau khi thực hiện sẽ thu được file MathService.cs

Tạo Web Service Proxy

- Thực hiện tạo file DLL từ dòng lệnh:
csc /t:library MathService.cs
- Chú ý: ứng dụng csc có thể tìm thấy tại thư mục cài đặt .NET\Framework, ví dụ:
C:\Windows\Microsoft.NET\Framework\v3.5
- Kết quả thu được là file MathService.dll

Tạo Web Service Proxy

- Tạo một Web Service Client:

```
using System;
```

```
class ServiceTest
```

```
{
```

```
    public static void Main (string[] argv)
```

```
    {
```

```
        MathService ms = new MathService();
```

```
        int x = Convert.ToInt16(argv[0]);
```

```
        int y = Convert.ToInt16(argv[1]);
```

```
        int sum = ms.Add(x, y);
```

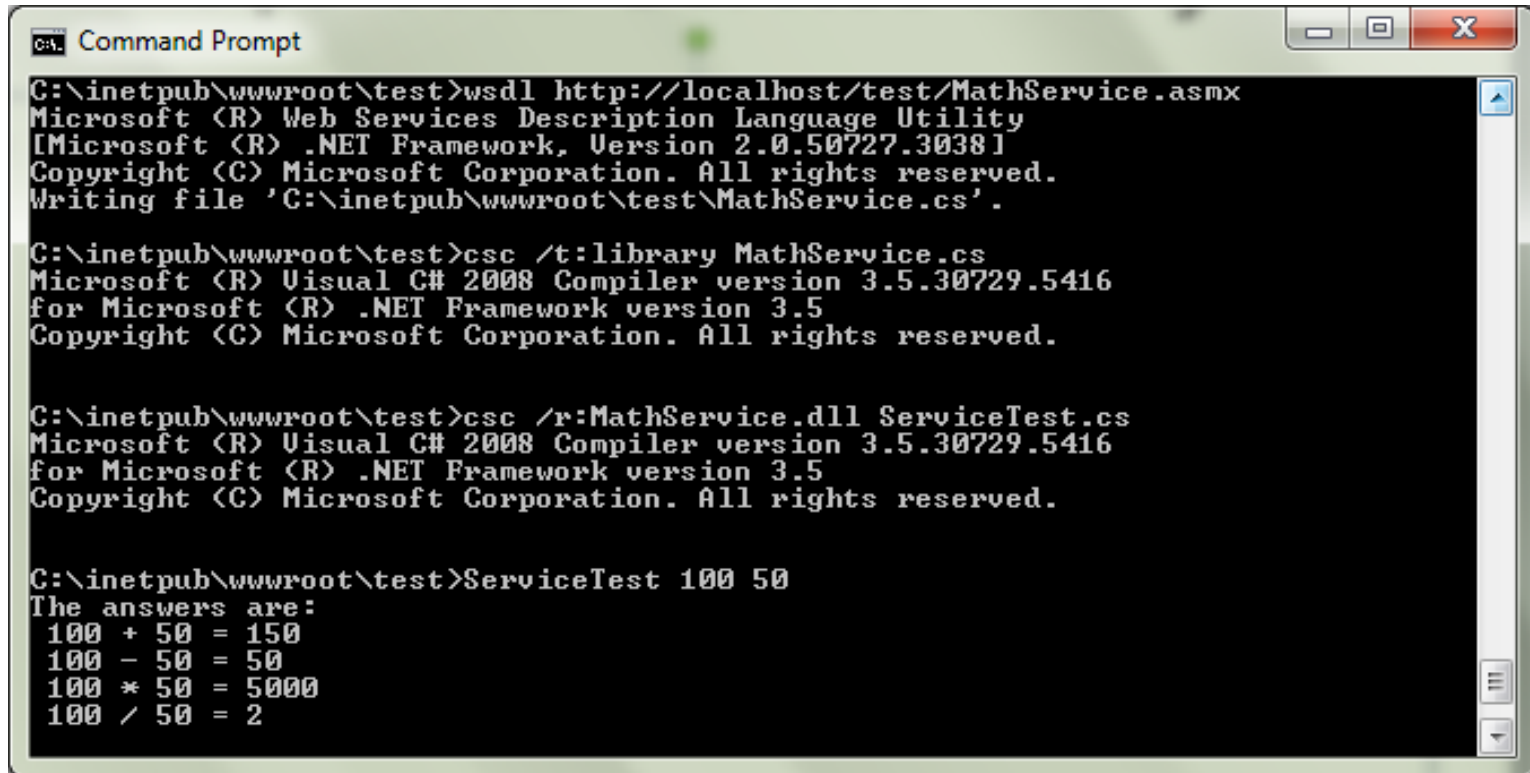

Tạo Web Service Proxy

```
int sub = ms.Subtract(x, y);  
int mult = ms.Multiply(x, y);  
int div = ms.Divide(x, y);  
Console.WriteLine("The answers are:");  
Console.WriteLine(" {0} + {1} = {2}", x, y, sum);  
Console.WriteLine(" {0} - {1} = {2}", x, y, sub);  
Console.WriteLine(" {0} * {1} = {2}", x, y, mult);  
Console.WriteLine(" {0} / {1} = {2}", x, y, div);  
}  
}
```

Tạo Web Service Proxy

- Thực hiện biên dịch chương trình client trên bằng dòng lệnh:
`csc /r:MathService.dll ServiceTest.cs`
- Kết quả thu được là file `ServiceTest.exe`
- Thực thi ứng dụng client với cú pháp:
`ServiceTest <số thứ 1> <số thứ 2>`
- Minh họa kết quả thực thi như hình sau:

Tạo Web Service Proxy



```
Command Prompt

C:\inetpub\wwwroot\test>wsdl http://localhost/test/MathService.asmx
Microsoft (R) Web Services Description Language Utility
[Microsoft (R) .NET Framework, Version 2.0.50727.3038]
Copyright (C) Microsoft Corporation. All rights reserved.
Writing file 'C:\inetpub\wwwroot\test\MathService.cs'.

C:\inetpub\wwwroot\test>csc /t:library MathService.cs
Microsoft (R) Visual C# 2008 Compiler version 3.5.30729.5416
for Microsoft (R) .NET Framework version 3.5
Copyright (C) Microsoft Corporation. All rights reserved.

C:\inetpub\wwwroot\test>csc /r:MathService.dll ServiceTest.cs
Microsoft (R) Visual C# 2008 Compiler version 3.5.30729.5416
for Microsoft (R) .NET Framework version 3.5
Copyright (C) Microsoft Corporation. All rights reserved.

C:\inetpub\wwwroot\test>ServiceTest 100 50
The answers are:
100 + 50 = 150
100 - 50 = 50
100 * 50 = 5000
100 / 50 = 2
```

Minh hoạ Web Service Proxy

- Khởi tạo project mới, kiểu ASP.NET Web Service.
- Đặt tên namespace MyServices
- Thêm code tương tự như class MathService (trong file MathService.asmx) ở ví dụ trước, chứa 4 phương thức Add, Subtract, Multiply, Divide
- Thực hiện project trên để khởi động server

Minh hoạ Web Service Proxy

- Xây dựng ứng dụng client
- Tạo project mới, kiểu Windows Form, gồm 1 form, 6 textbox có tên lần lượt là tbNumA, tbNumB, tbAdd, tbSubtract, tbMultiply, tbDivide; 1 button với tên btnCalculate
- Chọn project → Add Service References → Advanced → và tiến hành chọn Add Web References đến MyServices (chứa MathService), chọn tên Web Service là [localhost](#)

Minh hoạ Web Service Proxy

- Chỉnh sửa Web Reference URL của localhost cho đúng với địa chỉ server đang chạy, giả sử là <http://localhost:6415>
- Thêm đoạn code xử lý sự kiện Click của button trên như sau:

Minh hoạ Web Service Proxy

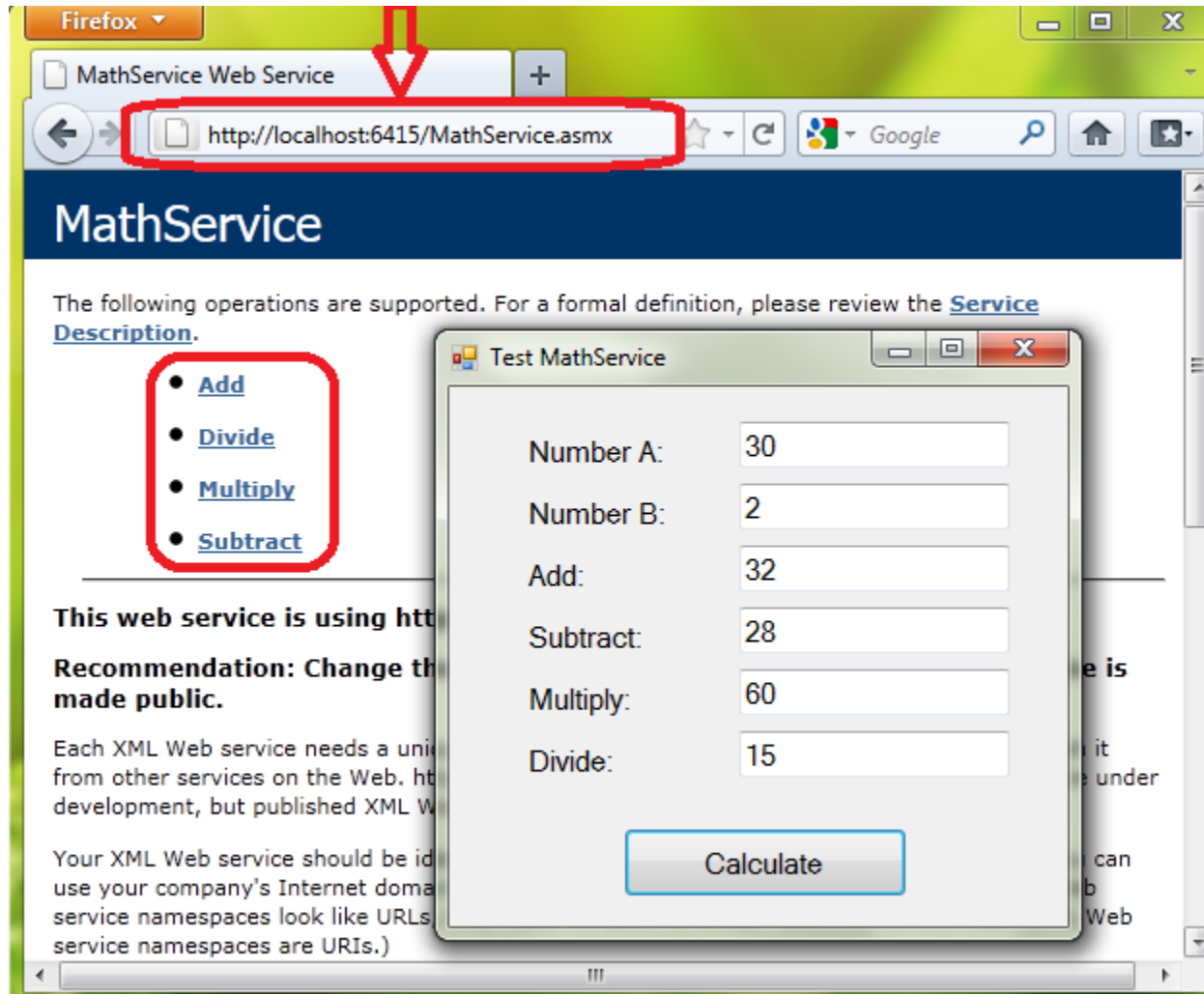
```
private void btnCalculate_Click(object sender, EventArgs e)
{
    localhost.MathService obj = new localhost.MathService();
    if (obj == null)
        System.Console.WriteLine("Could not locate localhost");
    else
    {
        int a = Convert.ToInt32(tbNumA.Text);
        int b = Convert.ToInt32(tbNumB.Text);
        int c = obj.Add(a, b);
    }
}
```

Minh hoạ Web Service Proxy

```
c = obj.Subtract(a, b);  
tbSubtract.Text = c.ToString();  
c = obj.Multiply(a, b);  
tbMultiply.Text = c.ToString();  
c = obj.Divide(a, b);  
tbDivide.Text = c.ToString();  
}
```

```
}
```


Minh hoạ Web Service Proxy

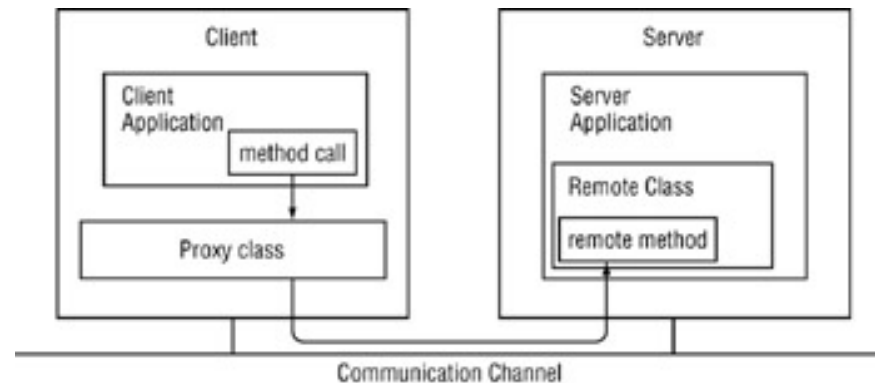


.NET remoting

- .NET remoting tương đương với Java Remote Method Invocation (RMI) và Distributed Common Object Model (DCOM) của Visual Basic
- Cho phép chia sẻ các class và phương thức giữa các máy tính trên mạng
- Giúp cho thuận tiện dùng các đối tượng phức tạp trên các máy tính ở xa
- Thuận lợi nữa là remoting trừu tượng hóa cơ sở hạ tầng mạng

.NET remoting

- Kiến trúc .NET remoting được mô tả như trong hình



- Không giống như Web service, remoting không cần cài thêm phần mềm server nào thêm để xử lý yêu cầu đến

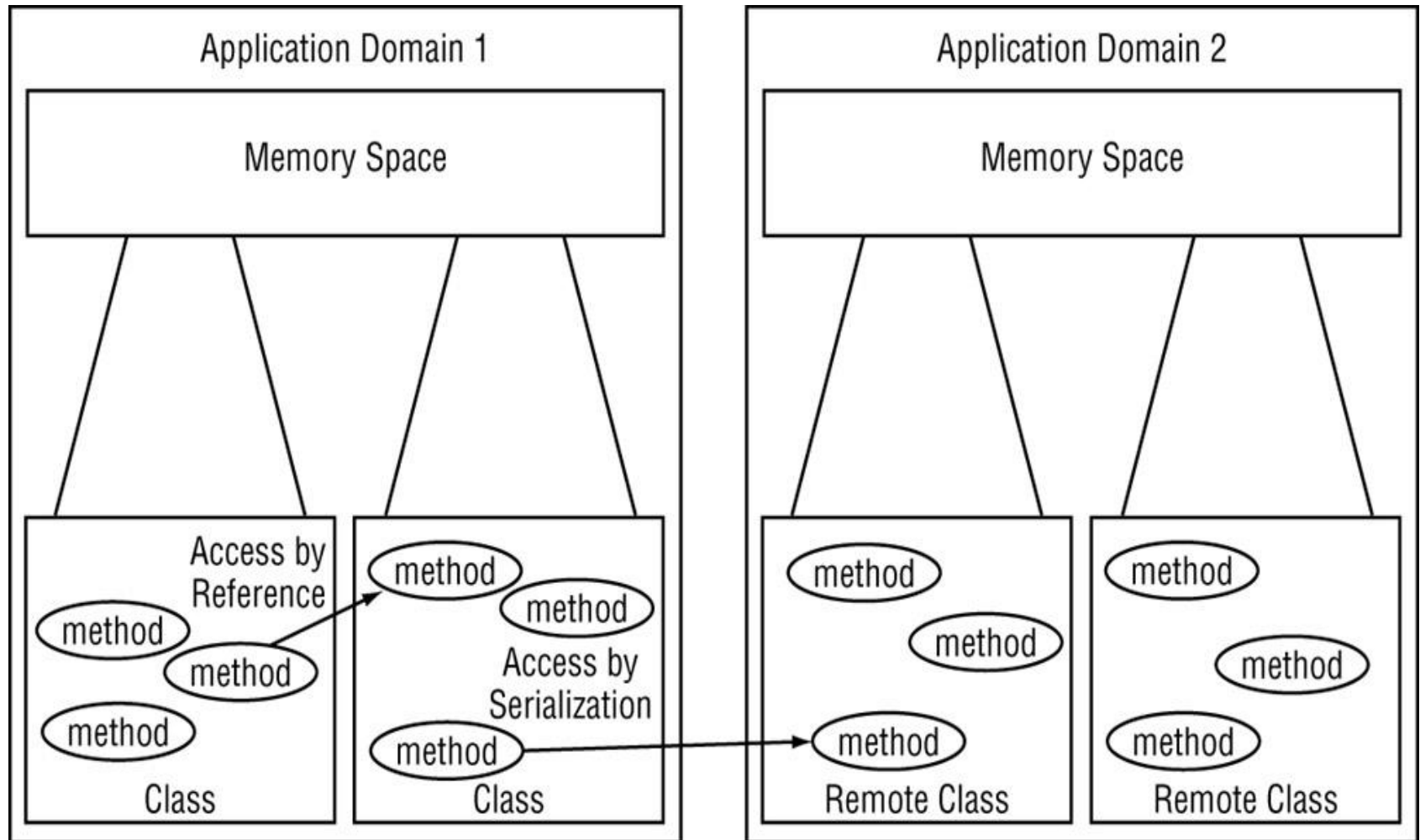
.NET remoting

- .NET remoting cho phép đơn giản hóa quá trình hiện thực các ứng dụng client/server trong đó server thực hiện một số tác vụ dưới sự chỉ dẫn của client
- Khi dùng remoting chúng ta cần tạo client và server. Đồng thời phải tạo một đối tượng thực hiện các chức năng theo yêu cầu

.NET remoting

- Một application domain là vùng mà tất cả các ứng dụng chia sẻ cùng một không gian bộ nhớ
- Bất kỳ class nào nằm ngoài application domain của một client đều được xem là một remote class
- Minh họa application domain của một client trong hình sau

.NET remoting



.NET remoting

- Khi các class nằm ngoài application domain thì chúng không thể truy xuất trực tiếp vào domain khác bằng tham chiếu (vì không có vùng nhớ chia sẻ). Thay vì vậy, mỗi thực thể của class phải được serialized và chuyển qua kênh truyền đến ứng dụng client khác

.NET remoting

- Các bên của kết nối đều phải biết kiểu của đối tượng, client cần biết IP và port của server.
- Mặc dù không nhìn thấy cái gì truyền trên mạng, nhưng chúng ta cần lựa chọn giữa SOAP trên HTTP (linh hoạt) hoặc binary trên TCP (hiệu suất)

.NET remoting

- SOAP cho remoting ít linh hoạt hơn Web service tương đương
- Để ngăn client làm kiệt quệ tài nguyên của server bởi việc tạo ra hàng triệu đối tượng và bỏ không dùng, remoting có một bộ thu gom rác tích hợp sẵn
- Các đối tượng có thể tạo nhưng vòng đời chỉ dài bằng thời gian thực thi của hàm (singlecall) hoặc class (singleton) hoặc vòng đời do server xác định (published objects)

.NET remoting

- Vòng đời của đối tượng remote (ngoại trừ published objects) được xác định nhờ việc gọi đến
`RemotingConfiguration.RegisterWellKnownService Type`
- Published objects được tạo ra như ví dụ:
`RemoteObject obj = new RemoteObject(1234);`
`RemotingServices.Marshal(obj,"RemotingServer");`

.NET remoting

- Sau đó đối tượng được hành động như một singleton. Thuận lợi của cách này là có thể tạo đối tượng với constructor không mặc định, cho phép người dùng can thiệp vào
- Điểm chính của remoting là tạo một lớp dẫn xuất từ MarshalByRefObject. Đối tượng này sau đó có thể chạy trong phạm vi server và trưng ra các phương thức, thuộc tính thông qua server đó

.NET remoting

- Trong khi chạy trong phạm vi server các tài nguyên local như file và cơ sở dữ liệu trên server được truy xuất thông qua class
- Các đối tượng được trả về như kết quả của phép gọi đến phương thức của class này, dù sao cũng chạy theo phạm vi client → chúng được gọi là các đối tượng By Value

.NET remoting

- Các đối tượng By Value không thể truy cập tài nguyên server như cơ sở dữ liệu hoặc file
- Một đối tượng remote trả về một By Value object bằng cách serializing và truyền nó trên mạng đến client. Để thực hiện được cần đáp ứng 2 điều kiện:
 1. Đối tượng phải đánh dấu bằng [Serializable] hoặc implement ISerializable
 2. Client phải quản lý metadata cho By Value object

Minh họa .NET remoting

- Minh họa ứng dụng remoting đơn giản, trong đó client nhận một số từ server, số này sẽ tăng 1 ở mỗi lần gọi đến
- Tạo 1 project Class library mới, thêm vào code ở slide sau
- Tiến hành compile để được file RemoteObject.DLL sẽ dùng sau này

Minh họa .NET remoting

```
using System;
namespace RemoteObject
{
    public class IDGenerator : System.MarshalByRefObject
    {
        private int lastID = 0;
        public int getID() {
            return(lastID++);
        }
    }
}
```

Minh họa .NET remoting

- Ứng dụng phía server
- Tạo 1 project Windows form mới, gồm 1 form
- Chọn Project→Add References→Browse để tham chiếu file RemoteObject.DLL
- Cũng cần tham chiếu thêm đến namespace System.Runtime.Remoting
- Thêm code xử lý sự kiện Load của form:

Minh họa .NET remoting

```
private void Form1_Load(object sender, EventArgs  
e)  
{  
    HttpChannel channel = new HttpChannel(8085);  
    ChannelServices.RegisterChannel(channel);  
    RemotingConfiguration.RegisterWellKnownServiceT  
ype(typeof(RemoteObject.IDGenerator),  
        "RemotingServer",  
        WellKnownObjectMode.Singleton);  
}
```

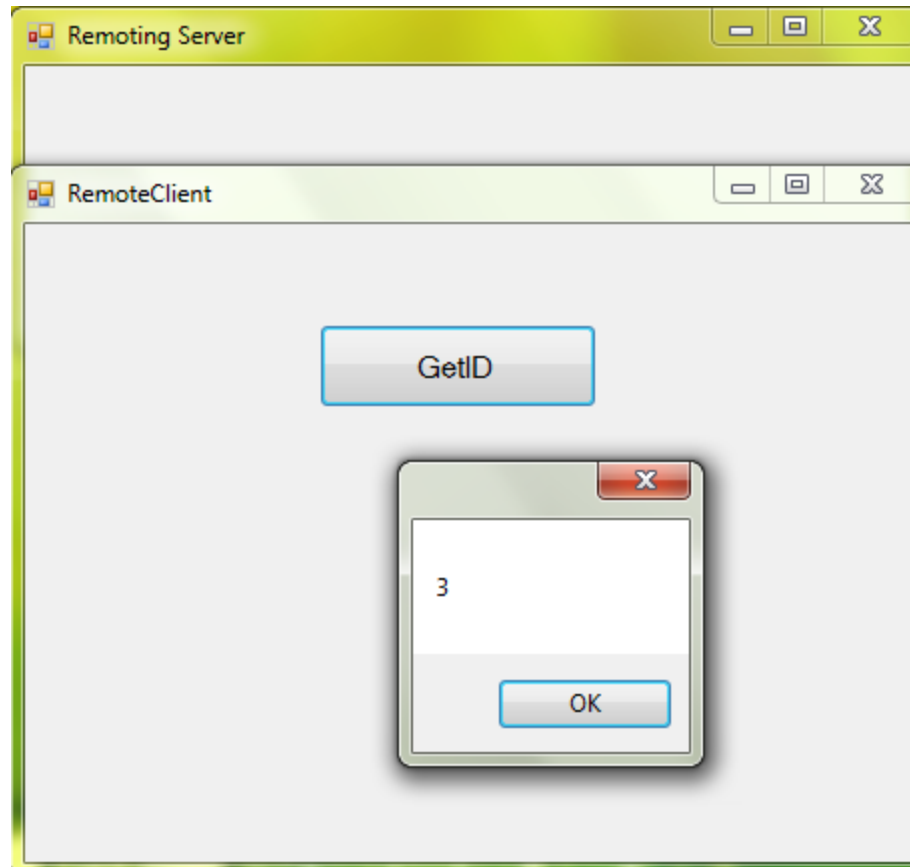
Minh họa .NET remoting

- Ứng dụng phía client
- Tạo 1 project Windows form mới, gồm 1 form, 1 button tên btnGetID
- Chọn Project→Add References→Browse để tham chiếu file RemoteObject.DLL
- Thêm code xử lý sự kiện Click của button btnGetID:

Minh họa .NET remoting

```
private void btnGetID_Click(object sender, EventArgs e)
{
    RemoteObject.IDGenerator remObject =
    (RemoteObject.IDGenerator)Activator.GetObject(
    typeof(RemoteObject.IDGenerator),
    "http://localhost:8085/RemotingServer");
    if (remObject == null)
        MessageBox.Show("Cannot locate server");
    else
        MessageBox.Show(Convert.ToString(remObject.getID()));
}
```

Minh họa .NET remoting



Minh họa .NET remoting

- Xây dựng ứng dụng minh họa thứ 2 về cơ chế remoting
- Tạo project mới, kiểu Class Library, đặt tên namespace là RemoteObject
- Khai báo class MathClass thừa kế từ class MarshalByRefObject như slide sau
- Biên dịch project trên để thu được file DLL tương ứng (sẽ dùng)

Minh họa .NET remoting

```
public class MathClass : MarshalByRefObject
{
    public int Add(int a, int b)
    {
        int c = a + b;
        return c;
    }
    public int Subtract(int a, int b)
    {
        int c = a - b;
        return c;
    }
}
```

Minh họa .NET remoting

```
public int Multiply(int a, int b)
{
    int c = a * b;
    return c;
}
public int Divide(int a, int b)
{
    int c;
    if (b != 0)
        c = a / b;
    else
        c = 0;
    return c;
}
}
```

Minh họa .NET remoting

- Xây dựng ứng dụng server
- Tạo project mới, kiểu Windows Form, gồm 1 form
- Chọn project → Add References và thêm 2 tham chiếu đến RemoteObject.DLL và System.Runtime.Remoting
- Thêm đoạn code xử lý sự kiện Load của form trên như sau:

Minh họa .NET remoting

```
private void Form1_Load(object sender, EventArgs
e)
{
    HttpChannel chan = new HttpChannel(9050);
    ChannelServices.RegisterChannel(chan);
    RemotingConfiguration.RegisterWellKnownServiceT
ype (typeof(RemoteObject.MathClass),
        "MyMathServer",
        WellKnownObjectMode.SingleCall);
}
```

Minh họa .NET remoting

- Xây dựng ứng dụng client
- Tạo project mới, kiểu Windows Form, gồm 1 form, 6 textbox có tên lần lượt là tbNumA, tbNumB, tbAdd, tbSubtract, tbMultiply, tbDivide; 1 button với tên btnCalculate
- Chọn project → Add References và thêm 2 tham chiếu đến RemoteObject.DLL và System.Runtime.Remoting
- Thêm đoạn code xử lý sự kiện Click của button trên như sau:

Minh họa .NET remoting

```
private void btnCalculate_Click(object sender,
EventArgs e)
{
    HttpChannel chan = new HttpChannel();
    ChannelServices.RegisterChannel(chan);
    MathClass obj = (MathClass)Activator.GetObject(
        typeof(MathClass),
        "http://127.0.0.1:9050/MyMathServer");
    if (obj == null)
        System.Console.WriteLine("Could not locate
server");
}
```

Minh họa .NET remoting

else

{

int a = Convert.ToInt32(tbNumA.Text);

int b = Convert.ToInt32(tbNumB.Text);

int c = obj.Add(a, b);

tbAdd.Text = c.ToString();

c = obj.Subtract(a, b);

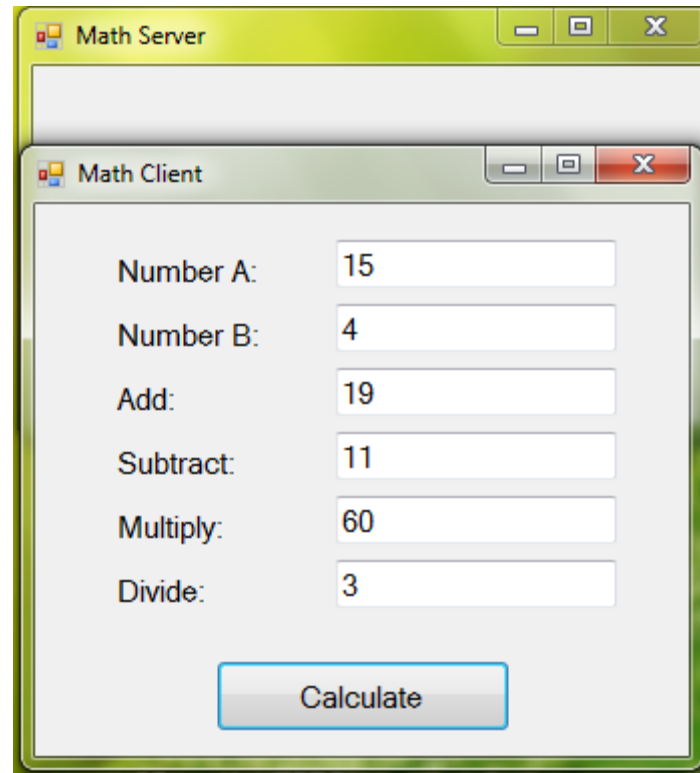
tbSubtract.Text = c.ToString();

Minh họa .NET remoting

```
c = obj.Multiply(a, b);  
tbMultiply.Text = c.ToString();  
c = obj.Divide(a, b);  
tbDivide.Text = c.ToString();  
}  
}
```

- Kết quả như hình minh họa ở slide sau

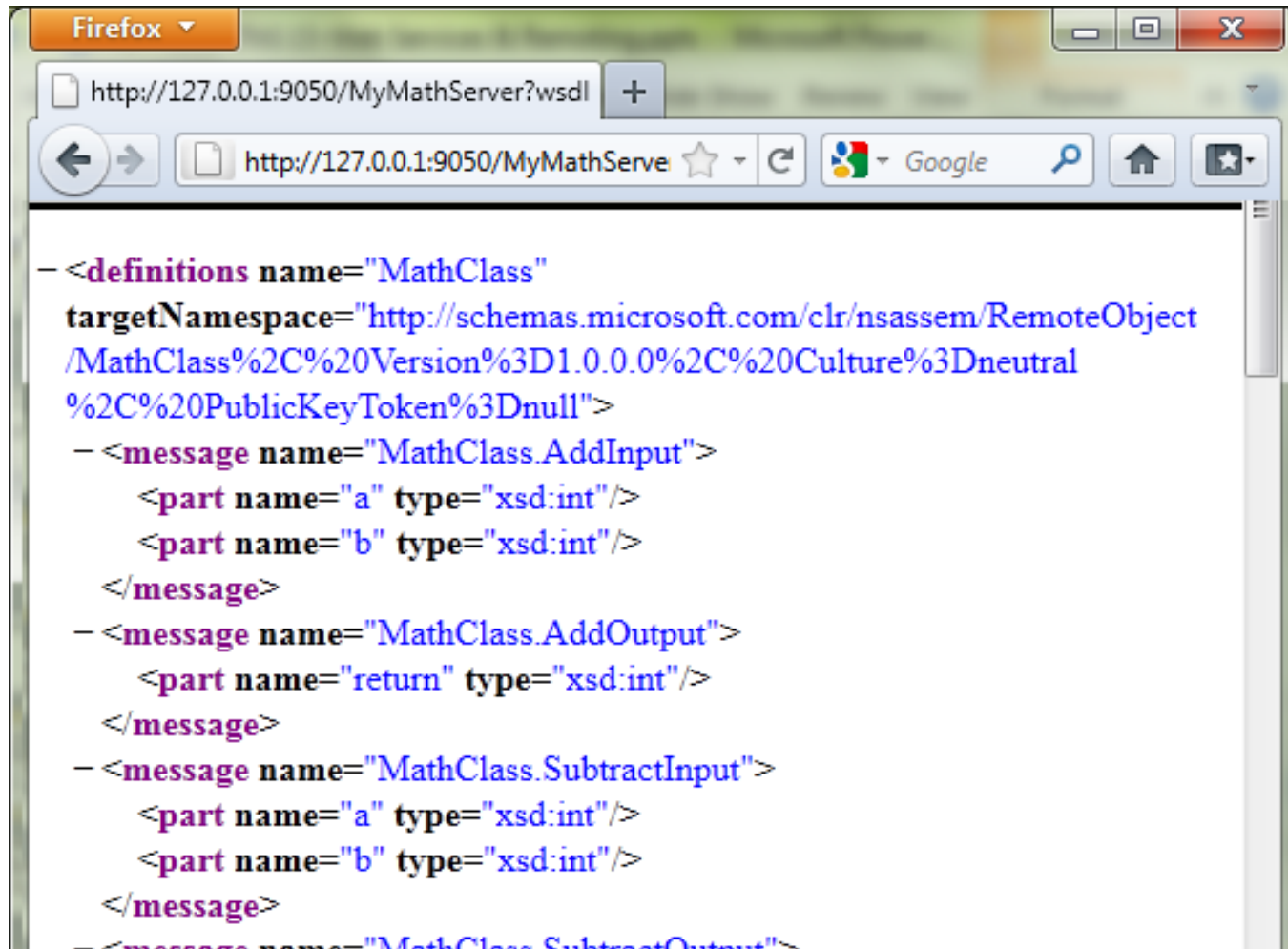
Minh họa .NET remoting



Xem remote class interfaces

- Khi HttpChannel được dùng để truyền thông với remoting server, định nghĩa của remote class có thể xem được thông qua một trình duyệt
- Để xem class interfaces, chúng ta thêm tag ?wsdl vào cuối URI. Ví dụ:
<http://127.0.0.1:9050/MyMathServer?wsdl>
- Kết quả như hình minh họa slide sau

Xem Remote Class Interfaces



The screenshot shows a Firefox browser window with the address bar displaying `http://127.0.0.1:9050/MyMathServer?wsdl`. The browser's address bar also shows a search bar with the text `http://127.0.0.1:9050/MyMathServe` and a Google search button. The main content area of the browser displays a WSDL document for a remote class interface named `MathClass`. The document is structured as follows:

```
- <definitions name="MathClass"
  targetNamespace="http://schemas.microsoft.com/clr/nsassem/RemoteObject
/MathClass%2C%20Version%3D1.0.0.0%2C%20Culture%3Dneutral
%2C%20PublicKeyToken%3Dnull">
  - <message name="MathClass.AddInput">
    <part name="a" type="xsd:int"/>
    <part name="b" type="xsd:int"/>
  </message>
  - <message name="MathClass.AddOutput">
    <part name="return" type="xsd:int"/>
  </message>
  - <message name="MathClass.SubtractInput">
    <part name="a" type="xsd:int"/>
    <part name="b" type="xsd:int"/>
  </message>
  - <message name="MathClass.SubtractOutput">
```


Dùng không đồng bộ các đối tượng remote

- Dùng không đồng bộ các đối tượng remote có thể thực hiện được nhờ cơ chế delegate (tương đương con trỏ hàm trong ngôn ngữ C++)
- Chúng được khai báo cùng phạm vi class với client, nhưng ngoài phạm vi đó với các phương thức của nó

Dùng không đồng bộ các đối tượng remote

- Có cùng kiểu khai báo prototype như phương thức đồng bộ chúng ta muốn gọi
- Ví dụ: Phương thức remote có tên getDetails() trả về string sẽ có delegate tương ứng khai báo và sử dụng như sau:

```
delegate String GetDetailsDelegate();  
GetDetailsDelegate gdDelegate = new  
GetDetailsDelegate(obj.GetDetails);  
IASyncResult gdAsyncres =  
gdDelegate.BeginInvoke(null,null);
```

Dùng không đồng bộ các đối tượng remote

- Server sẽ thực thi `getDetails()` trên đối tượng remote. Để lấy giá trị trả về từ việc gọi hàm, client phải thực thi `EndInvoke` trên delegate. Phương thức này sẽ blocking và chỉ trả về một khi server đã đáp ứng.
- Việc gọi được thực hiện như sau:
`String details = gdDelegate.EndInvoke(gnAsyncre);`

Dùng không đồng bộ các đối tượng remote

- Cũng có một cách khác để kích hoạt một đối tượng remote không đồng bộ, dùng thuộc tính `OneWay`.
- Gọi One-way được tạo giống như cách gọi chuẩn từ client, ngoại trừ `EndInvoke` sẽ không blocking và chắc chắn trả về ngay lập tức không cần biết server đã đáp ứng hay chưa

Dùng không đồng bộ các đối tượng remote

- Phương pháp trên được áp dụng cho nơi yêu cầu tốc độ và không quan tâm lắm đến các vấn đề khác
- Để hiện thực hàm one-way, đơn giản đánh dấu phương thức bên trong phần định nghĩa giao tiếp với thuộc tính [OneWay()]

Triển khai một remoting service

- Khi dùng remoting trong một ứng dụng thương mại, có một số thủ thuật để giúp cho phần mềm được mạnh mẽ và dễ quản lý hơn.
- Client phải có khả năng nêu loại đối tượng nó muốn nhận tại thời điểm biên dịch. Nghĩa là khi đã triển khai phần mềm cho hàng triệu người dùng, chúng ta không thể

Triển khai một remoting service

- thực hiện các thay đổi đến các đối tượng đó hoặc bắt buộc các client dừng hoạt động. Cách giải quyết là cho client tham chiếu đến interface của đối tượng chứ không phải chính đối tượng. Như vậy khi thay đổi hiện thực của phương thức của đối tượng thì không cần ngắt ngang các hoạt động liên quan đến đối tượng đó

Triển khai một remoting service

- Một khía cạnh quan trọng nữa là chúng ta có thể chia sẻ phần hiện thực của phần mềm với bên thứ 3

- Một interface cho class

RemoteObject.IDGenerator trên như sau:

```
using System;  
public Interface IIDGenerator  
{  
    public int getID();  
}
```


Tạo một proxy class dùng soapsuds

- Các ví dụ trong chương này giả định một tình huống quan trọng là chúng ta đang điều khiển cả remoting server và client.
- Thực tế điều này không phải lúc nào cũng có được
- Chúng ta sẽ nghiên cứu cách tạo một ứng dụng client ngay cả khi không có remote class dùng để tạo client

Tạo một proxy class dùng soapsuds

- .NET Framework SDK có chương trình
soapsuds giúp khai thác thông tin class từ
remote class (xem thêm chương trình wsdl)

Bài tập

- Cài đặt các chương trình đã minh họa trong bài giảng của chương bằng ngôn ngữ C# hoặc VB.NET