

مقرر: برمجة التطبيقات الشبكية

الفصل الثالث: خدمات الويب Web Services

فهرس الفصل:

1. مقدمة.
2. خدمات الويب.
3. البروتوكول البسيط للوصول للأغراض.
4. نشر واستهلاك خدمات الويب.
5. استخدام خدمات الويب في نماذج الويب.
6. الأنماط المعرفة في خدمات الويب.
7. الخلاصة.

1. مقدمة

يستخدم المبرمجون عادةً مكتبات الربط الديناميكية (DLLs) **Dynamic Link Libraries** لتسهيل عملية إعادة استخدام البرمجيات. إلا أن هذه المكتبات يجب أن توضع على نفس الحاسب الذي يستخدم أو يستدعي هذه البرمجيات.

ندرس في هذا الفصل استخدام تقانات خدمات الويب للسماح بإعادة استعمار البرمجيات في الأنظمة الموزعة. تسمح تقانات الأنظمة الموزعة بتنفيذ التطبيقات على عدة حواسيب ضمن شبكة.

تسمح خدمة ويب **Web Service** بالقيام بالحسابات الموزعة وذلك من خلال السماح لحاسب من استدعاء طريقة **method** موجودة على حاسب آخر عبر صيغ بيانات مشتركة وبروتوكولات معيارية مثل **XML** و **HTTP**.

تُنفَّذ عمليات الاستدعاء عبر الشبكة للطرق في **NET**. باستخدام بروتوكول الوصول للأغراض البسيط **SOAP**. والذي هو بروتوكول مبني على **XML** ويقوم بتوصيف كيفية تأشير **mark up** الطلبات **requests** والأجوبة **responses** وبحيث يتم تناقلها عبر بروتوكولات معيارية مثل **HTTP**.

يمكن استخدام **SOAP** التطبيقات من تمثيل وتناقل البيانات في قوالب معيارية **XML**. تُشجع **Microsoft** مسوقي البرمجيات والأعمال الالكترونية على نشر وتشارك خدمات الويب. إذ طالما تتزايد يوماً بعد يوم أعداد المتصلين بالانترنت، فإن استدعاء التطبيقات للطرق عبر الانترنت يصبح مربحاً.

ومن منظور البرمجة غرضية التوجه، تُمثل خدمات الويب الجيل الثاني من البرمجة غرضية التوجه. إذ عوضاً عن برمجة مجموعة قليلة من الصفوف في مكان واحد، سيصبح المبرمجون قادرين على استخدام عدد هائل من الصفوف تتواجد في أماكن متفرقة متباعدة.

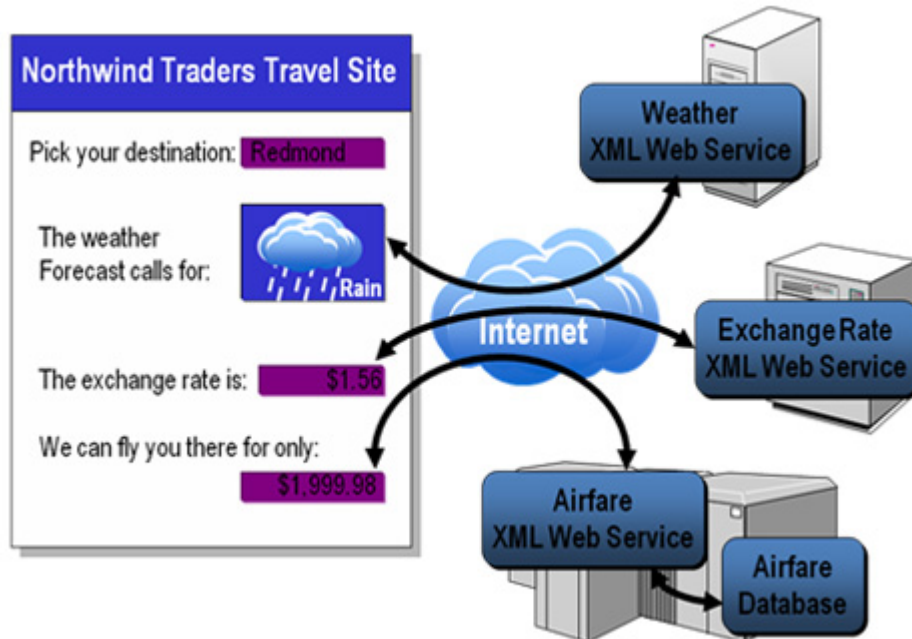
تسمح هذه التقنية بتسهيل عملية تعاون تطبيقات الأعمال الضخمة ونموها بشكل سريع. إذ بالتشارك في خدمات الويب الموافقة، ستمكن الشركات العاملة في تطوير المنتجات البرمجية من اختصار الزمن اللازم للتطوير والاتفات دائماً لتطوير خدمات جديدة.

لنأخذ كمثال بسيط، حالة شركة تباع على الويب أقراص موسيقية كما تُعطى معلومات عن الفنانين. ولنفرض أن شركة أخرى تختص ببيع بطاقات الحفلات تُقدم خدمة ويب تسمح بمعرفة تواريخ الحفلات القادمة لمختلف الفنانين وتسمح للمستخدمين بشراء بطاقات هذه الحفلات. يمكن للشركتين أن تتعاوننا وبحيث يتمكن مستخدمي موقع الشركة الأولى من شراء بطاقات الحفلات مما سيزيد من عدد زوار موقع الشركة الأولى.

كذلك فإن الشركة الثانية ستزداد مبيعاتها إذ أنها ستستفيد من زبائن الشركة الأولى كما يمكن لها أن تطالب عمولة من الشركة الأولى على البطاقات المباعة.

يمثل الشكل التالي مثلاً صفحة ويب لشركة سفر وسياحة تستفيد من عدة خدمات ويب لمساعدة زبونها:

- خدمة ويب لمعرفة حالة الطقس للوجهة المسافرين إليها.
- خدمة ويب لمعرفة سعر الصرف للعملة المحلية.
- خدمة ويب لمعرفة ثمن التذكرة.



2. خدمات الويب Web Services

تكون خدمة الويب عبارة عن تطبيق موجود على حاسب وبحيث يمكن لأي حاسب آخر على الشبكة من الوصول إلى هذا التطبيق. يدعى عادة الحاسب الحاضن للتطبيق بالحاسب البعيد **remote machine**.

يقوم التطبيق الذي يريد الاستفادة من خدمة الويب بإرسال طلب استدعاء لطريقة توفرها خدمة الويب للحاسب البعيد، والذي يقوم بدوره بمعالجة هذا الطلب وإرسال الجواب إلى التطبيق المستدعي.

يفيد هذا النمط من الحسابات الموزعة العديد من الأنظمة، بما فيها تلك الأنظمة التي لا يمكن لها الوصول إلى بيانات معينة أو تلك التي لاتملك قدرات كافية للقيام ببعض الحسابات المعينة. إذ أنها ستتمكن عبر استخدامها لخدمات الويب من الاستفادة من مصادر حواسيب ضخمة.

سنرى أن خدمة الويب هي في النهاية صف **Class** يحوي مجموعة من الطرق **Methods** ويتواجد على حاسب بعيد.

يتمّ نقل الطلبات والأجوبة باستخدام البروتوكول **SOAP**. مما يعني أن أي زبون قادر على توليد ومعالجة رسائل **SOAP** يستطيع استخدام خدمة الويب، وذلك بغض النظر عن اللغة التي كُتبت بها خدمة الويب. كما يبين الشكل التالي:



يمكن لخدمة الويب أن تفحص خدماتها على زبائن مخولين **authorized** فقط.

وبما أن خدمات الويب والبروتوكول SOAP مستقلين عن منصات التشغيل واللغات المستخدمة، فيمكن للشركات أن تتعاون إذاً عبر خدمات الويب دون أن تعلق لتوافقية البرمجيات أو العتاد أو تقانات الاتصال.

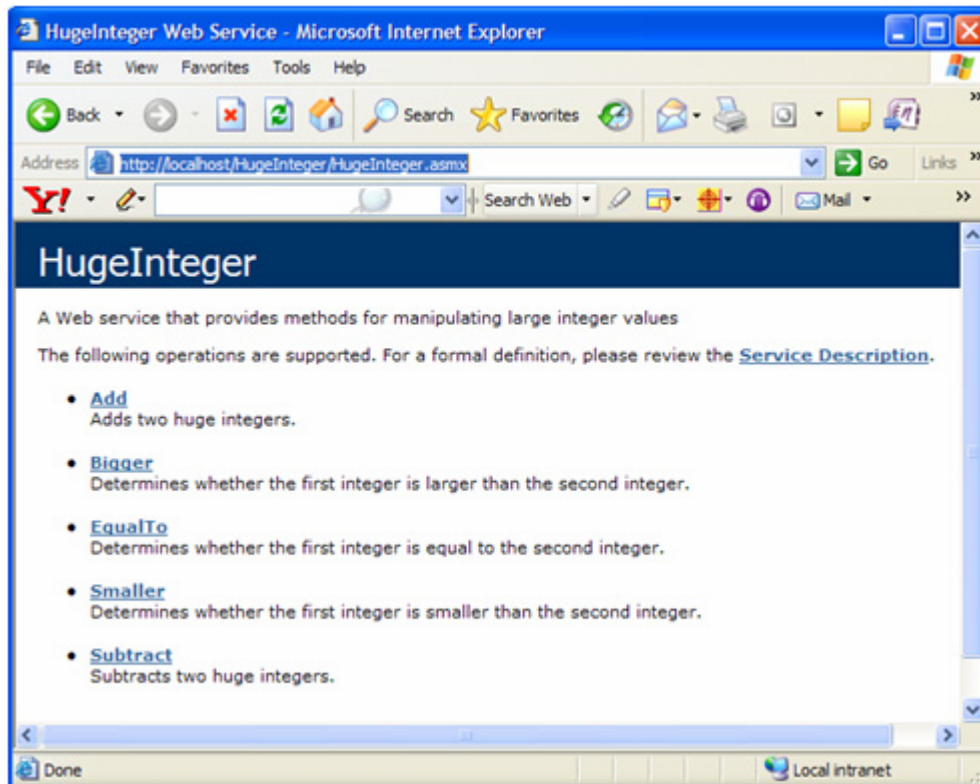
تتعاون مثلاً الشركات مثل Amazon و Google و eBay عبر تشارك مجموعة م-ن خ-د-مات الويب.

تُستدعى الطرق الموجودة في خدمة ويب عن بعد باستخدام إجرائية الاستدعاء البعيدة Remote Procedure Call (RPC). يتمّ تأشير هذه الطرق باستخدام الوصفة WebMethod كما سي-نرى لاحقاً.

يجب لإنشاء خدمة ويب باستخدام Visual Web Developer أن تنشئ أولاً مشروع م-ن-ز-وع ASP.NET Web Service والذي سيولد مايلي:

- الملفات التي ستضم كود خدمة الويب (مختلف الطرق المؤلفة للخدمة).
- ملف asmx والذي يؤمن الوصول إلى خدمة الويب.

يبين الشكل التالي مثلاً مايعيده استدعاء ملف asmx من قبل المتصفح:



تحتوي الصفحة ارتباطات إلى مختلف الطرق المتاحة في خدمة الويب. يمكن بهـالنقر على أي ارتباط لطريقة فتح الصفحة التي تسمح بإدخال معاملات هذه الطريقة. مثلاً لو أردنا اختبار الطريقة **EqualTo**:

HugeInteger Web Service - Microsoft Internet Explorer

File Edit View Favorites Tools Help

Back Forward Stop Home Search Favorites Reload Print

Address <http://localhost/HugeInteger/HugeInteger.asmx?op=EqualTo> Go Links

Search Web

HugeInteger

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

EqualTo

Determines whether the first integer is equal to the second integer.

Test

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

Parameter	Value
first:	123456789
second:	987654321

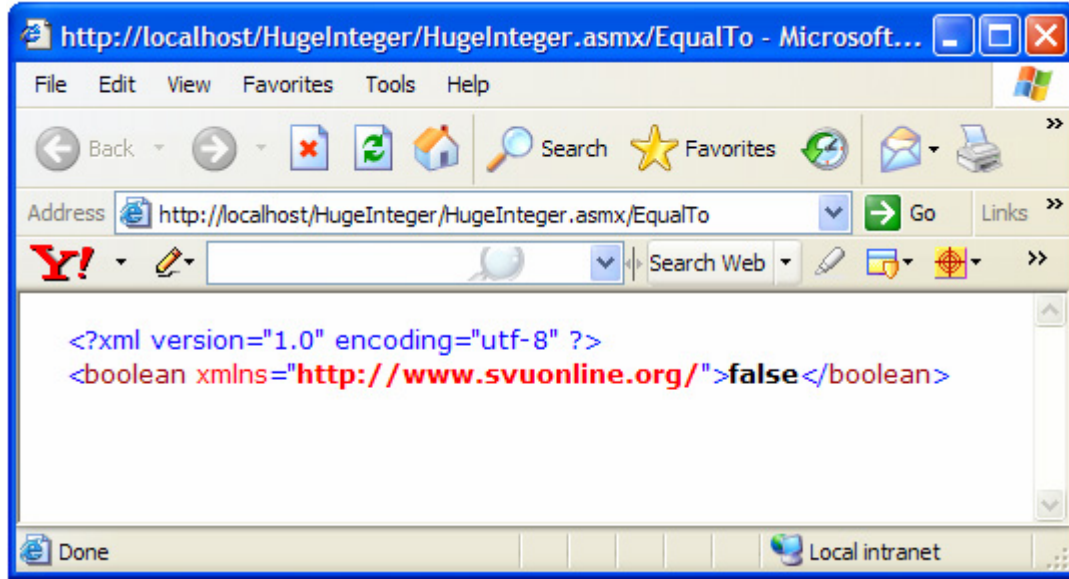
Invoke

SOAP 1.1

The following is a sample SOAP 1.1 request and response. The **placeholders** shown need to be replaced with actual values.

```
POST /HugeInteger/HugeInteger.asmx HTTP/1.1
Host: localhost
Content-Type: text/xml; charset=utf-8
```

وبالنقر على الزر **Invoke** يظهر لنا نتائج تنفيذ الطريقة:



يحتوي رأس الصفحة الأولى لخدمة الويب الارتباط **Service Description** والذي يفـتح ماـف توصيف الخدمة. يكون ماـف توصيف الخدمة عبارة عن ماـف XML مطـابق للغة المعيارية المستخدمة لتوصيف خدمات الويب (**Web Services Description Language (WSDL)**). والتي هي لغة XML تُستخدم لتوصيف الطرق المتاحة في خدمة ويب وكيفية التعامل مع هذه الطرق.

3. البروتوكول البسيط للوصول للأغراض Simple Object Access Protocol (SOAP)

يمكن تعريف البروتوكول البسيط للوصول للأغراض SOAP بأنه بروتوكول مستقل عن منصة التشغيل يستخدم XML لإنشاء الاستدعاءات البعيدة للإجراءات عبر البروتوكول HTTP.

يحزم كل طلب **request** وجواب **response** في رسالة SOAP أي رسالة XML تحوي المعلومات اللازمة لمعالجة محتواها.

لقد تم اختيار البروتوكول HTTP لإرسال رسائل SOAP، لأنه بروتوكول معياري لتراسل البيانات على الإنترنت.

يمكن استخدام كل من HTTP و XML نظام التشغيل المختلفة من إرسال واستقبال رسائل SOAP. كما أن من الميزات الإضافية الأخرى للبروتوكول HTTP هو أنه يمكن استخدامه في الشبكات التي تحوي جدران نارية **fire walls** تُقيد الاتصالات في الشبكة. إذ أن هذه الجدران الأمنية لا تحد عملياً من حركة HTTP.

يدعم البروتوكول SOAP مجموعة كبيرة من أنماط البيانات Data Types. إذ يجب على صيغة النقل wire format المستخدمة لتبادل الطلبات والأجوبة أن تدعم جميع أنماط البيانات المستخدمة في التطبيقات.

تدعم خدمات الويب التي تستخدم SOAP أنماط بيانات أكثر من خدمات الويب التي تستخدم صيغاً أخرى للنقل.

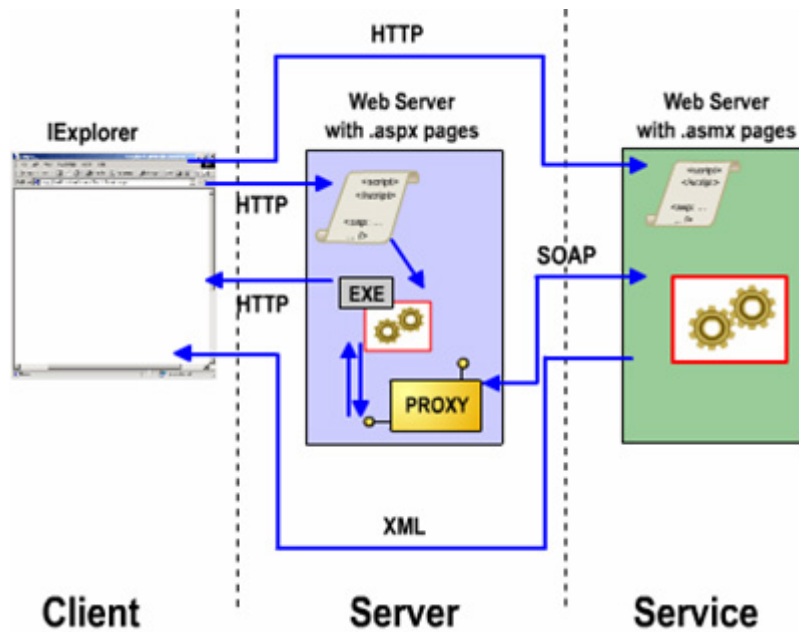
تتضمن أنماط البيانات في SOAP جميع أنماط المعطيات الأساسية (مثل integer, double, . . .)، كذلك تتضمن الأنماط المركبة (مثل DataSet, DateTime, XmlNode). كما يسمح البروتوكول SOAP بتبادل مصفوفات arrays من البيانات من جميع الأنماط. كذلك يمكن استخدام الأنماط المعروفة من قبل المستخدم user-defined types.

تقوم التطبيقات بإرسال الطلبات والأجوبة إلى ومن خدمات الويب عبر رسائل SOAP. عندما يستدعي برنامج طريقة من خدمة ويب، فإن الطلب وجميع المعلومات المتعلقة به تُحزم في رسالة SOAP، وتُرسل إلى الجهة الموافقة. عندما تستقبل خدمة الويب الرسالة تبدأ بمعالجة محتواها (المدعو بالمغلف envelope) والذي يحدد الطريقة التي يريد الزبون تنفيذها والمعاملات parameters الواجب تمريرها إلى هذه الطريقة. تُستدعى بعد ذلك الطريقة مع المعاملات الموافقة ويتم إرسال ناتج تنفيذ الطريقة إلى الزبون في رسالة SOAP أخرى. وم الزبون بمعالجة هذه الرسالة للوصول إلى نتائج الطريقة.

يتألف التطبيق الذي يستهلك خدمة ويب عملياً من قسمين: صف وكيـل proxy يمثل خدمة الويب، وتطبيق زبون يصل إلى خدمة الويب من خلال غرض من الصف السابق.

يعهد إلى صف الوكيل مهمة نقل المعاملات لطريقة خدمة الويب من التطبيق الزبون إلى خدمة الويب، كما يقوم بنقل نتائج طريقة خدمة الويب إلى التطبيق الزبون. يقوم Visual Studio بتوليد صفوف proxy كما سوف نرى.

يبين الشكل التالي مخططات العمليات السابقة:



4. نشر واستهلاك خدمات الويب Publishing and Consuming Web Services

وتشمل:

1. تعريف خدمة الويب
2. إنشاء خدمة ويب.
3. نشر خدمة الويب.
4. إنشاء زيون لاستهلاك خدمة الويب.
5. استهلاك خدمة ويب.

تعريف خدمة الويب

نعرض فيما يلي الكود الخلفي لخدمة الويب المثال. صممت هذه الخدمة للقيام بالحسابات على أعداد صحيحة تحوي على الأكثر 100 خانة. (لا يمكن للمتحويلات من النوع long أن تصل لهذا الحجم). تُقدم خدمة الويب الطرق التالية والتي تأخذ كل منها رقمين صحيحين ضخمين كمعاملتي دخل:

- **Bigger** لاختبار أن العدد الضخم الأول أكبر من العدد الضخم الثاني.
- **EqualTo** لاختبار أن العددين الضخمين متساويين.
- **Smaller** لاختبار أن العدد الضخم الأول أصغر من العدد الضخم الثاني.
- **Add** لجمع العددين الضخمين.

• **Subtract** اطرح العدد الضخم الثاني من الأول.

يمكن لأي تطبيق الوصول إلى هذه الخدمة عبر الويب والاستفادة من الطرق المتوفرة فيها - دون أن يضطر المبرمج لكتابة الكود اللازم لهذه الطرق:

```
1 // HugeInteger.cs
2 // HugeInteger Web service performs operations on large integers.
3 using System;
4 using System.Web;
5 using System.Web.Services;
6 using System.Web.Services.Protocols;
7
8 [WebService(Namespace = "http://www.svuonline.org/",
9     Description = "A Web service that provides methods for" +
10     " manipulating large integer values")]
11 [WebServiceBinding(ConformsTo = WsiProfiles.BasicProfile1_1)]
12 public class HugeInteger : System.Web.Services.WebService
13 {
14     private const int MAXIMUM = 100; // maximum number of digits
15     public int[] number; // array representing the huge integer
16
17     // default constructor
18     public HugeInteger()
19     {
20         number = new int[MAXIMUM];
21     } // end default constructor
22
23     // indexer that accepts an integer parameter
24     public int this[int index]
25     {
26         get
27         {
28             return number[index];
29         } // end get
30
31         set
32         {
33             number[index] = value;
34         } // end set
35     } // end indexer
36
37     // returns string representation of HugeInteger
38     public override string ToString()
39     {
40         string returnString = "";
41
42         foreach (int i in number)
43             returnString = i + returnString;
44
45         return returnString;
46     } // end method ToString
47
48     // creates HugeInteger based on argument
```

```

49     public static HugeInteger FromString(string value)
50     {
51         // create temporary HugeInteger to be returned by the method
52         HugeInteger parsedInteger = new HugeInteger();
53
54         for (int i = 0; i < value.Length; i++)
55             parsedInteger[i] = Int32.Parse(
56                 value[value.Length - i - 1].ToString());
57
58         return parsedInteger;
59     } // end method FromString
60
61 // WebMethod that adds integers represented by the String arguments
62 [WebMethod(Description = "Adds two huge integers.")]
63 public string Add(string first, string second)
64 {
65     int carry = 0;
66     HugeInteger operand1 = HugeInteger.FromString(first);
67     HugeInteger operand2 = HugeInteger.FromString(second);
68     HugeInteger result = new HugeInteger();
69
70     // perform addition algorithm for each digit
71     for (int i = 0; i < MAXIMUM; i++)
72     {
73         // add two digits in same column,
74         // result is their sum plus carry from
75         // previous operation, modulo 10
76         result[i] =
77             (operand1[i] + operand2[i] + carry) % 10;
78
79         // set carry to remainder of dividing sums of two digits by 10
80         carry = (operand1[i] + operand2[i] + carry) / 10;
81     } // end for
82
83     return result.ToString();
84 } // end method Add
85
86 // WebMethod that subtracts integers
87 // represented by the string arguments
88 [WebMethod(Description = "Subtracts two huge integers.")]
89 public string Subtract(string first, string second)
90 {
91     HugeInteger operand1 = HugeInteger.FromString(first);
92     HugeInteger operand2 = HugeInteger.FromString(second);
93     HugeInteger result = new HugeInteger();
94
95     // subtract bottom digit from top digit
96     for (int i = 0; i < MAXIMUM; i++)
97     {
98         // if top digit is smaller than bottom digit we need to borrow
99         if (operand1[i] < operand2[i])
100             Borrow(operand1, i);
101
102         // subtract bottom from top
103         result[i] = operand1[i] - operand2[i];
104     } // end for
105
106     return result.ToString();
107 } // end method Subtract

```

```

108
109     // borrow 1 from next digit
110     private void Borrow(HugeInteger hugeInteger, int place)
111     {
112         // if no place to borrow from, signal problem
113         if (place >= MAXIMUM - 1)
114             throw new ArgumentException();
115
116         // otherwise if next digit is zero, borrow from column to left
117         else if (hugeInteger[place + 1] == 0)
118             Borrow(hugeInteger, place + 1);
119
120         // add ten to current place because we borrowed and subtract
121         // one from previous digit--this is the digit we borrowed from
122         hugeInteger[place] += 10;
123         hugeInteger[place + 1]--;
124     } // end method Borrow
125
126     // returns true if first integer is bigger than second
127     [WebMethod(Description = " whether the first integer is " +
128         "larger than the second integer.")]
129     public bool Bigger(string first, string second)
130     {
131         char[] zeros = { '0' };
132
133         try
134         {
135             // if elimination of all zeros from result
136             // of subtraction is an empty string,
137             // numbers are equal, so return false, otherwise return true
138             if (Subtract(first, second).Trim(zeros) == "")
139                 return false;
140             else
141                 return true;
142         } // end try
143         // if ArgumentException occurs, first
144         // number was smaller, so return false
145         catch (ArgumentException exception)
146         {
147             return false;
148         } // end catch
149     } // end method Bigger
150
151     // returns true if first integer is smaller than second
152     [WebMethod(Description = " whether the first integer " +
153         "is smaller than the second integer.")]
154     public bool Smaller(string first, string second)
155     {
156
157         return Bigger(second, first);
158     } // end method Smaller
159
160     // WebMethod that returns true if two integers are equal
161     [WebMethod(Description = " whether the first integer " +
162         "is equal to the second integer.")]
163     public bool EqualTo(string first, string second)
164     {
165         // if either first is bigger than second, or first is
166         // smaller than second, they are not equal

```

```

167         if (Bigger(first, second) || Smaller(first, second))
168             return false;
169         else
170             return true;
171     } // end method EqualTo
172 } // end class HugeInteger

```

تحتوي الأسطر 8-11 الوصفة **WebService**. تضم هذه الوصفة الخصائص التالية:

- **Namespace** فضاء العمل: وأعطيناها القيمة <http://www.svuonline.org/>.
- **Description** لتوصيف خدمة الويب. يعرض الملف **asmx** هذا الوصف.
- **WebServiceBinding** تُضاف تلقائياً وتُحدد معيار خدمة الويب.

يشتق كل صف من خدمة ويب جديد تلقائياً الأسطر (12) من الصف **System.Web.Services.WebService** والذي يحوي طرق مفيدة لتحديد معلومات عن الزبون وعن خدمة الويب نفسها.

تتمّ تأشير العديد من الطرق في الصف بالوصفة **WebMethod** لتحديد بأن هذه الطرق هي الطرق المتاحة في خدمة الويب الأسطر (62, 88, 127, 152, 161) أي يمكن استدعائها عن بعد. عندما لا توضع هذه الوصفة لطريقة، لا تكون هذه الطريقة متاحة للزبائن عن بعد. تحوي هذه الوصفة أيضاً الخاصية **Description** والتي ستسمح للملف **asmx** بعرض وصف للطريقة.

تُعرّف الأسطر 24-35 مفرس **indexer** يمكن من الوصول لأي خلية في **HugeInteger**.
تُعرّف الأسطر 62-84 طريقة الويب **Add**. كما تُعرّف الأسطر 88-107 طريقة الويب **Subtract**.

تُعرّف الأسطر 110-124 طريقة **Borrow** (ليست طريقة ويب) والتي تُعالج حالة طرح خانة أكبر من خانة أصغر.

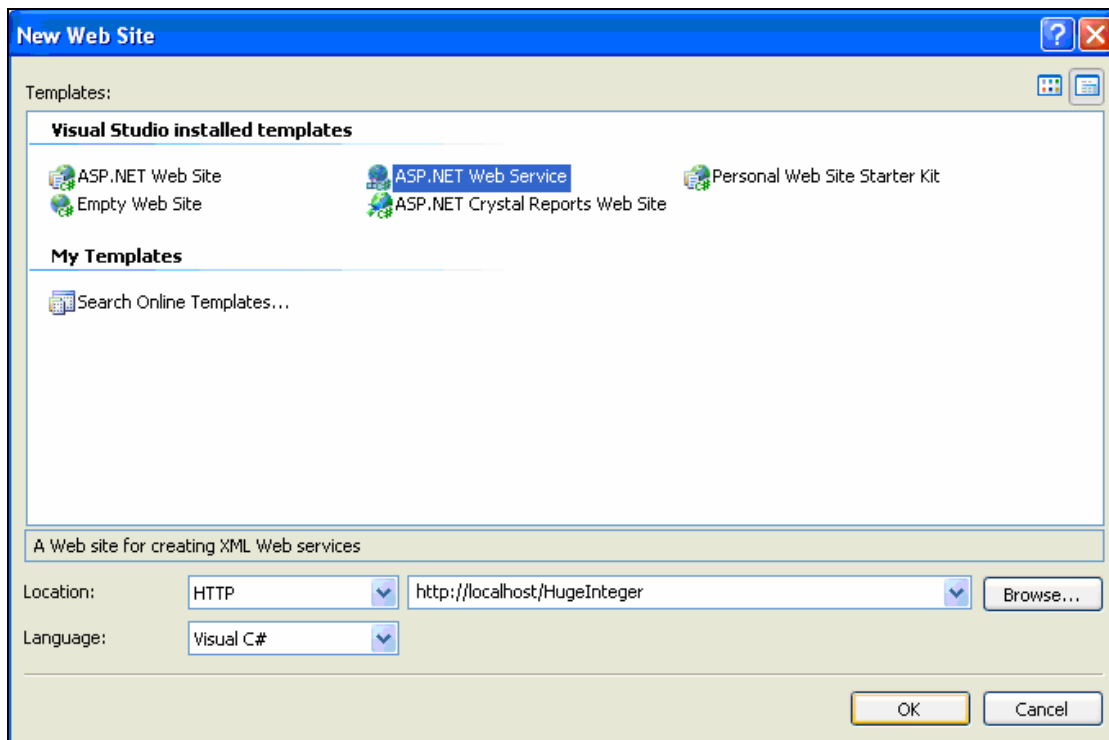
(مثلاً، عندما نطرح 19 من 32 نجد أن الرقم 2 هو أصغر من 9 وبالتالي فإننا نضيف 10 إلى 2 مما يعطينا 12 ، بعد الاستعارة نستطيع طرح 9 من 12 ويكون الناتج هو 3 ، ثم نطرح 1 من 3). (2)

إنشاء خدمة ويب

تُبين الخطوات التالية الخطوات اللازمة لإنشاء خدمة الويب بـ **HugeInteger** باستخدام **Visual Studio** على مخدم الإنترنت المحلي **Internet Information Services IIS**:

1. الخطوة الأولى: إنشاء المشروع.

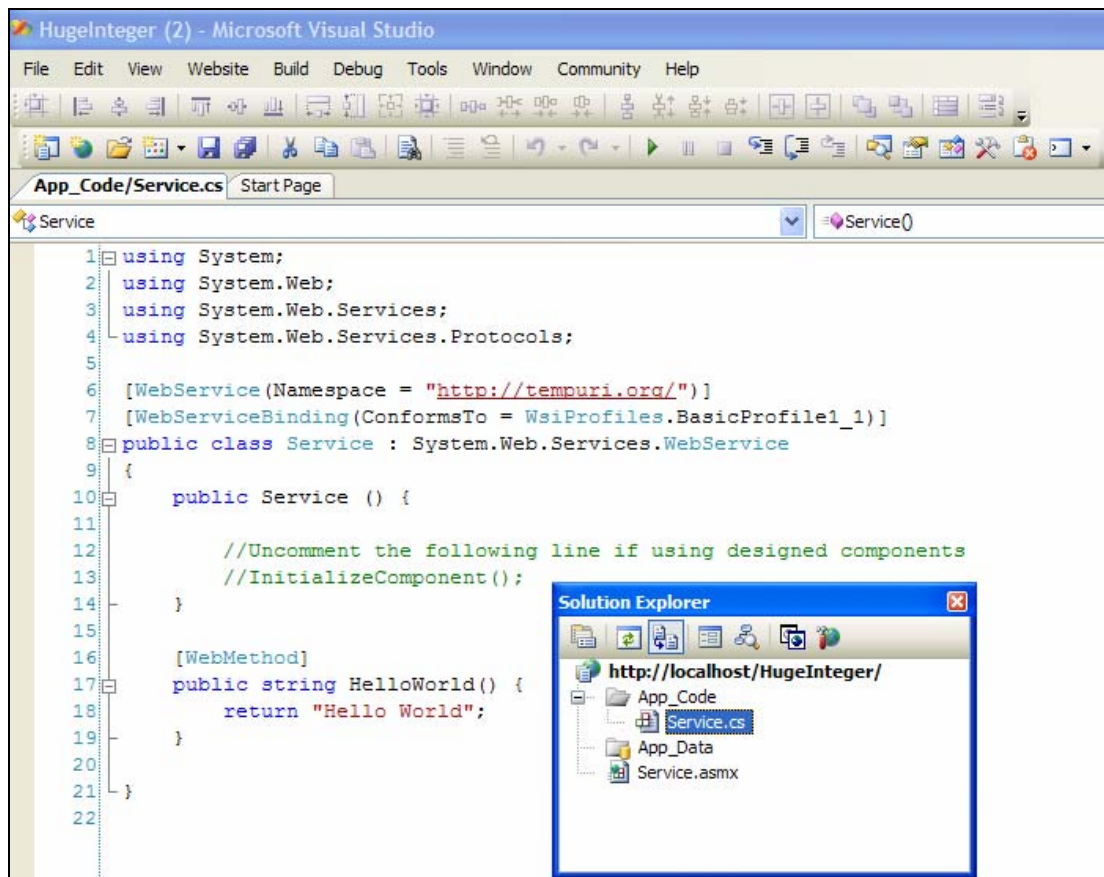
- اختيار **File → New Web Site**.
- اختيار القالب **ASP.NET Web Service**.
- اختيار **Location: HTTP**.
- العنوان **<http://localhost/HugeInteger>**.
- اللغة **Visual C#**.



2. الخطوة الثانية: فحص الخدمة الأولية المنشأة:

يقوم **Visual Studio** بإنشاء خدمة ويب بسيطة تتألف مما يلي:

- الصف **Service.cs** الذي يحوي الطريقة **HelloWorld**، وهو صف **ف-م-م** مشتق من الصف **System.Web.Services.WebService**.



• الملف Service.asmx والذي يحوي:

```

<%@ WebService Language="C#"
CodeBehind="~/App_Code/Service.cs" Class="Service" %>

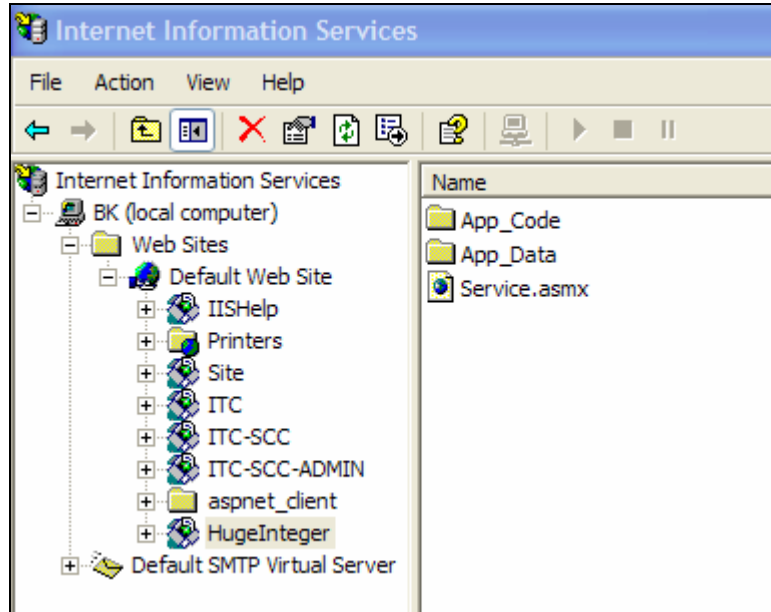
```

محدداً لغة البرمجة المستخدمة، مسار الكود الخلفي، واسم الصف.

(عند طلب الملف `asmx`. عبر IIS كما سنرى، تُستخدم هذه المعلومات لتوليد الناتج)

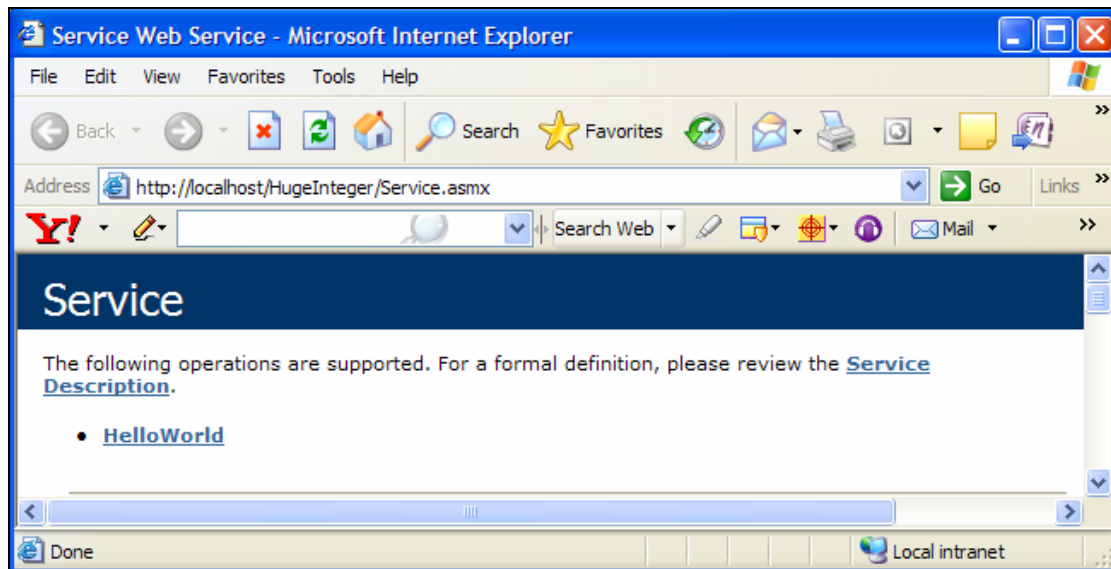
3. الخطوة الثالثة: التأكد من استضافة خدمة الويب

يمكن فتح IIS للتأكد من استضافة خدمة الويب المنشأة على الحاسب كما يبين الشكل:

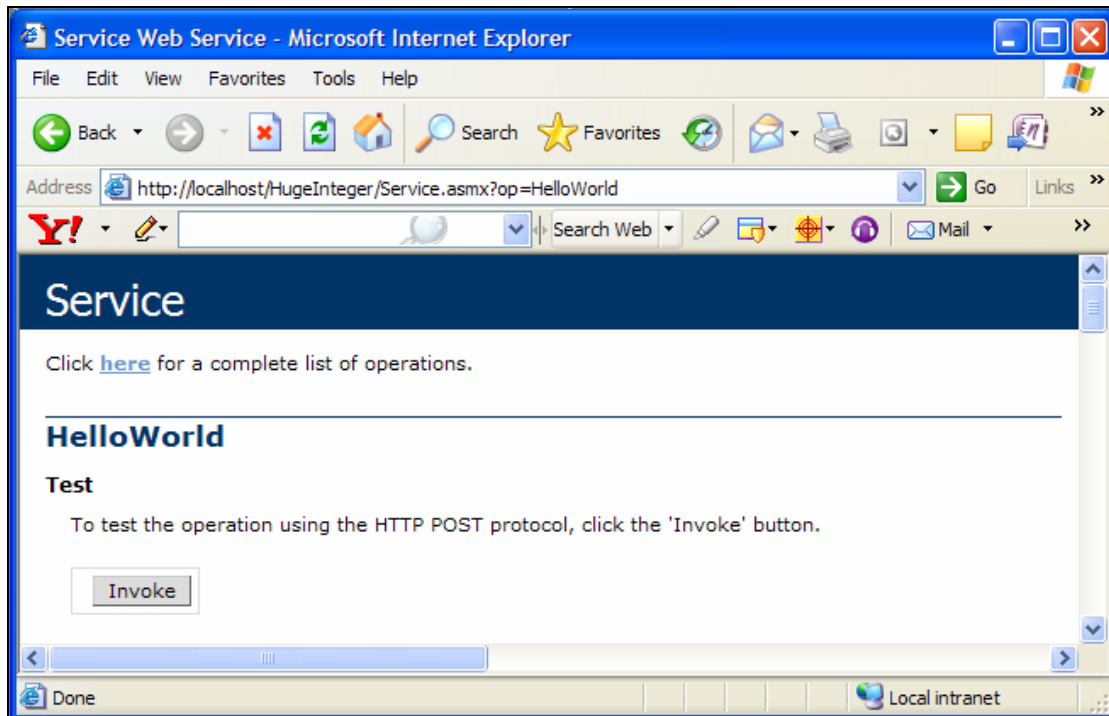


4. الخطوة الرابعة: تجريب خدمة الويب

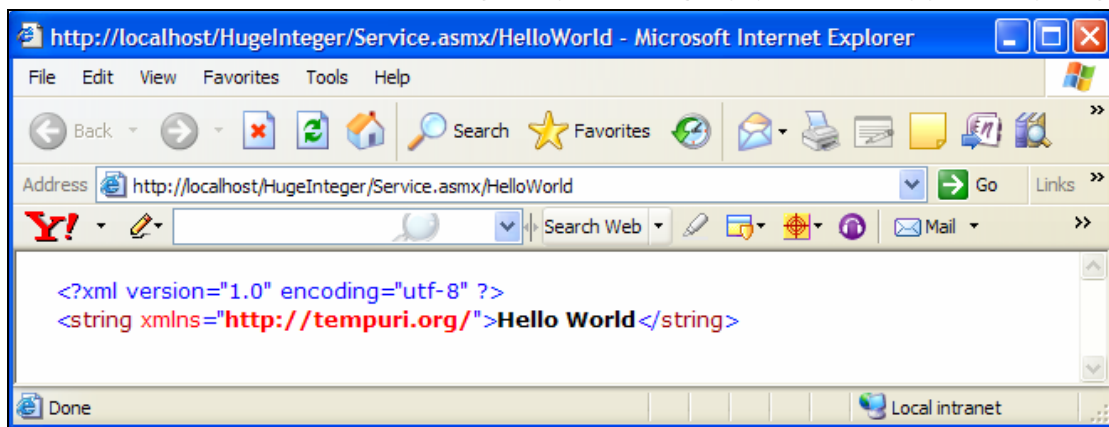
يمكن فتح المستعرض الآن مع العنوان `http://localhost/HugeInteger/Service.asmx` لتشغيل خدمة الويب:



وبالنقر على الارتباط HelloWorld تظهر النافذة التالية:



وبالنقر على الزر Invoke يظهر ناتج تنفيذ طريقة الويب:



5. الخطوة الخامسة: تعديل الخدمة الأولية إلى الخدمة المطلوبة:

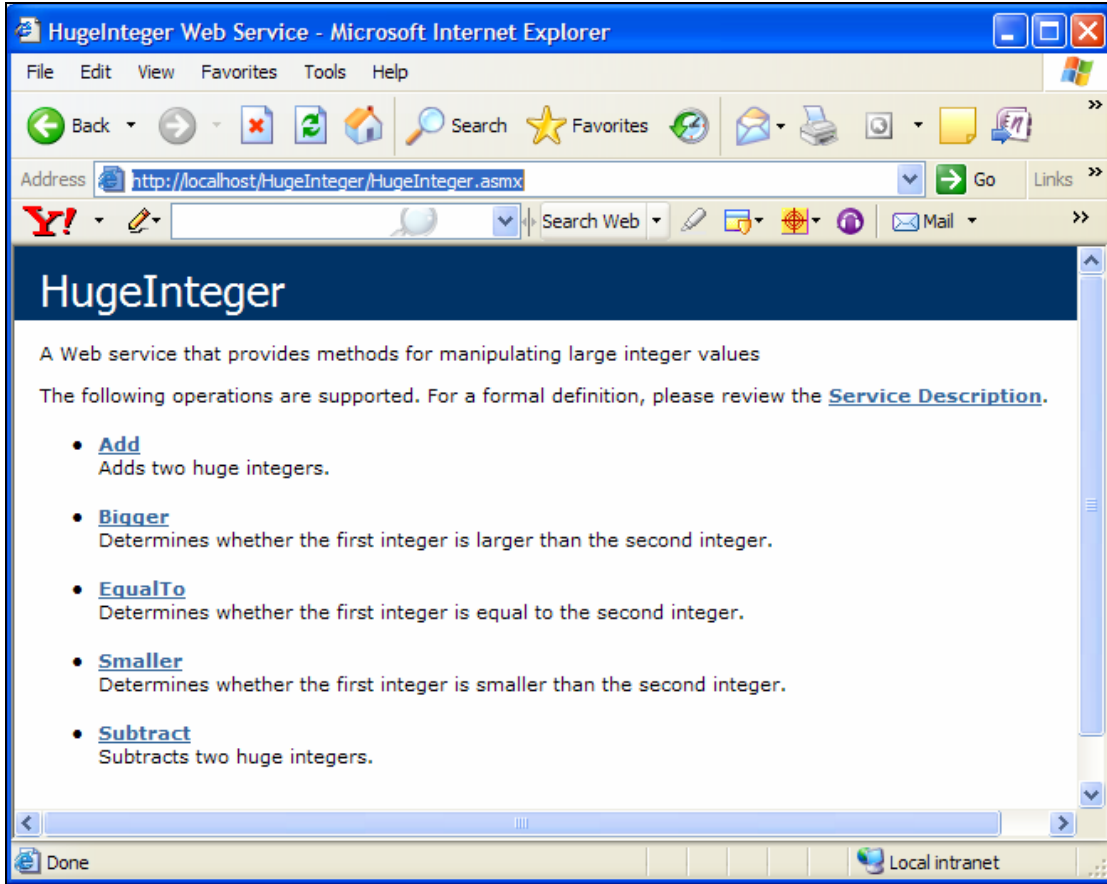
تتضمن هذه الخطوة:

- كتابة الكود المطلوب للخدمة HugeInteger (راجع الفقرة السابقة).
- تعديل اسم الملف Service.cs إلى HugeInteger.cs.
- تعديل اسم الملف Service.aspx إلى HugeInteger.aspx.
- تعديل محتوى الملف HugeInteger.aspx إلى:

```
<%@ WebService Language="C#" CodeBehind="~/App_Code/HugeInteger.cs"
Class="HugeInteger" %>
```

6. الخطوة السادسة: تجريب خدمة الويب من المستعرض

يمكن الآن طلب الخدمة HogeInteger من المـستعرض على العـنـوان <http://localhost/HogeInteger/HogeInteger.aspx> كما يبين الشكل:



لنطلب مثلاً الطريقة Add:

نشر خدمة الويب

تكون خدمة الويب السابقة HugelInteger منشورة deployed تلقائياً لأننا أنشأناها على IIS مباشرة. أي يمكن فتح خدمة الويب من المستعرض كما وضعنا سابقاً مع العنوان:

`http://localhost/HugelInteger/HugelInteger.asmx`

يمكن في Visual Studio اختيار Build Web Site → Build للتأكد من خلـو المـشروع مـن الأخطاء.

كما يمكن تجريب خدمة الويب ضمن Visual Studio باختبار Debug → Start Debugging مما سيؤدي لفتح المستعرض للملف .asmx.

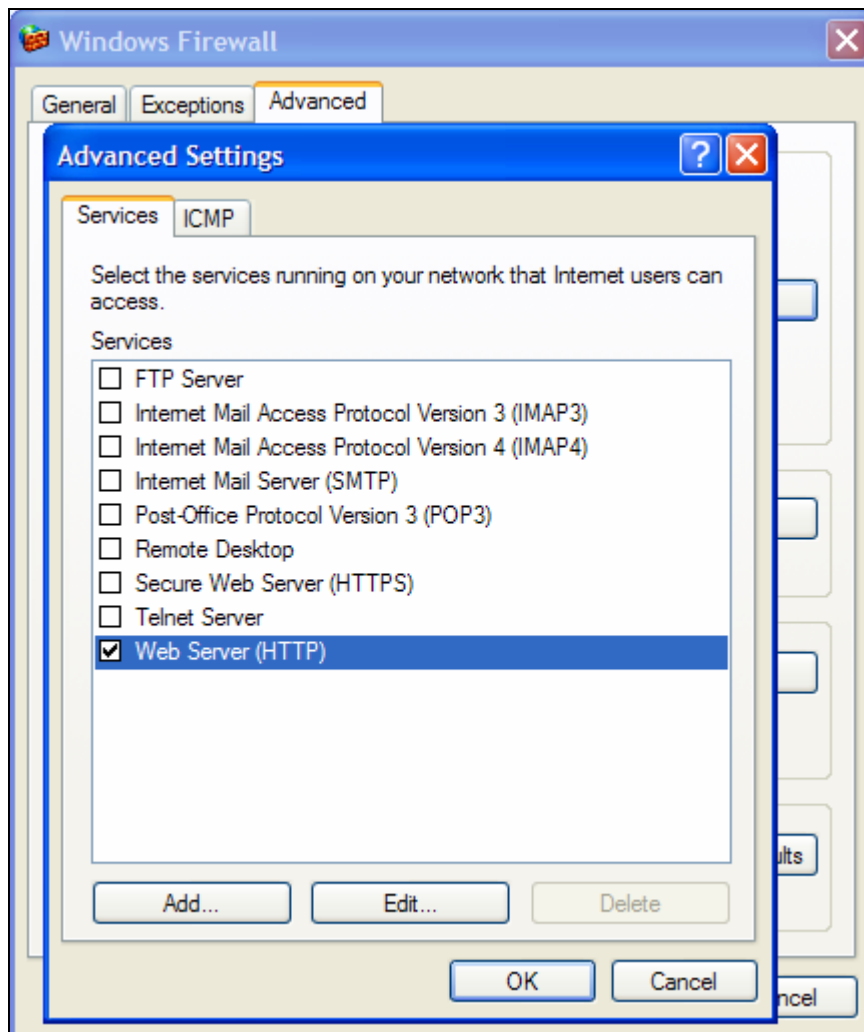
يمكن للآخرين الموصولين عبر الشبكة من استخدام خدمة الويب المنشورة على IIS على حاسبك بكتابة العنوان التالي في المستعرض:

`http://host/HugelInteger/HugelInteger.asmx`

حيث host هو اسم حاسبك أو عنوانك IP address على الشبكة.

يجب القيام بالخطوات التالية للسماح للآخرين باستخدام خدمات الويب من حاسبك:

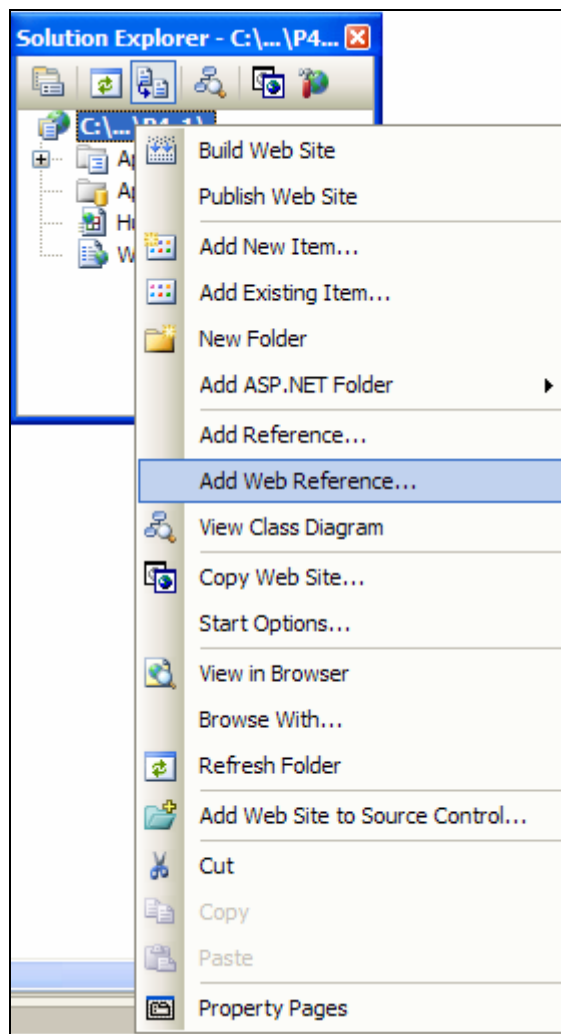
1. اختر Start → Control Panel لفتح نافذة التحكم للحاسب، ثم اختر Windows Firewall.
2. افتح الصفحة Advanced في نافذة Windows Firewall.
3. انقر الزر Setting في الصفحة Advanced.
4. اختر صندوق التحقق (HTTP) Web Server.



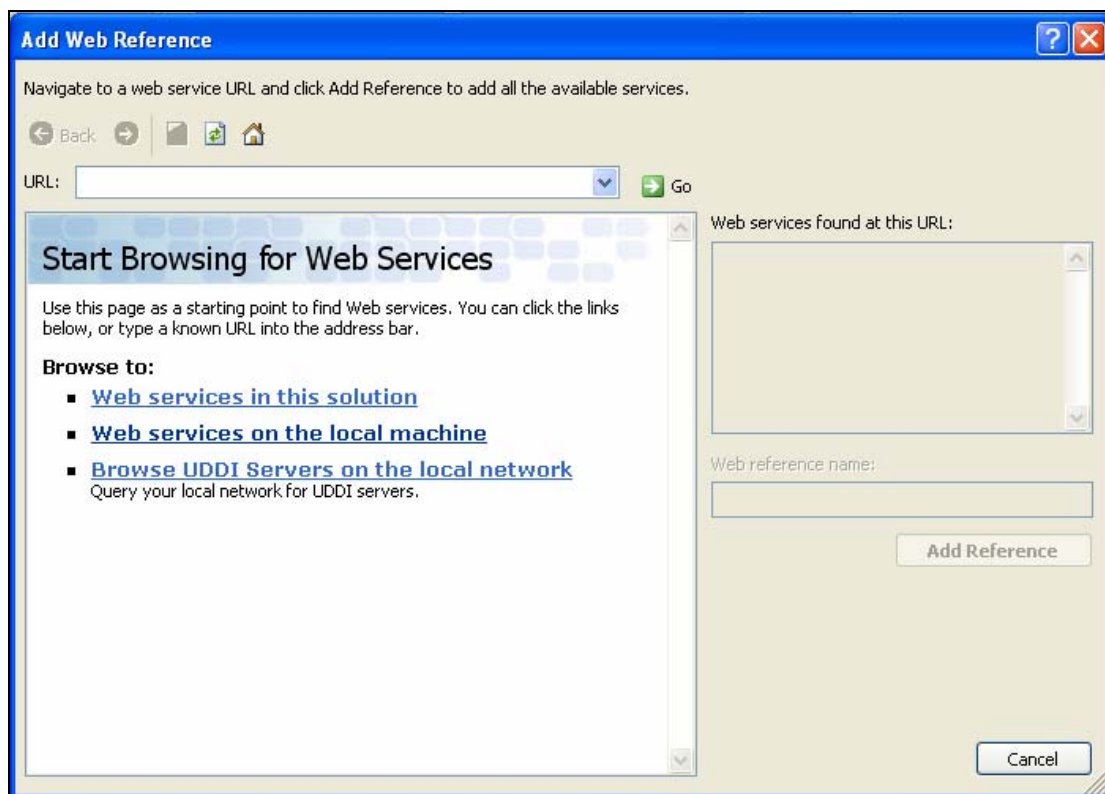
إنشاء زبون لاستهلاك خدمة الويب

يمكن الآن، بعد القيام بتعريف ونشر خدمة الويب، إنشاء زبون يستهلك هذه الخدمة.
يجب أولاً إنشاء مشروع من النمط **Windows Application** ثم القيام بالخطوات التالية:

1. الخطوة الأولى: فتح صندوق الحوار **Add Web Reference** يتم ذلك بفتح **Solution Explorer** ومن ثم **Add Web Reference** من قائمة الزر الأيمن:

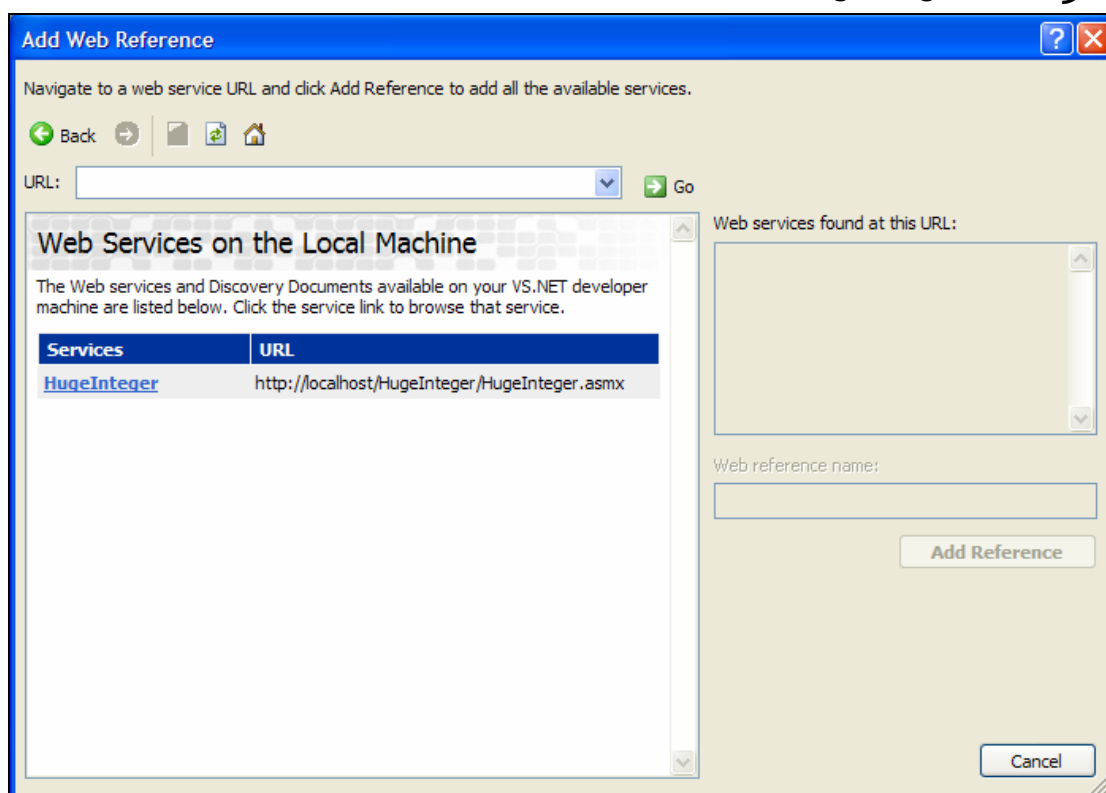


2. الخطوة الثانية: تحديد خدمات الويب المتوفرة على الحاسب
اختر **Web services on the local machine** لإظهار مراجع الويب المستضافة على الحاسب.
المحلي في IIS (<http://localhost>).



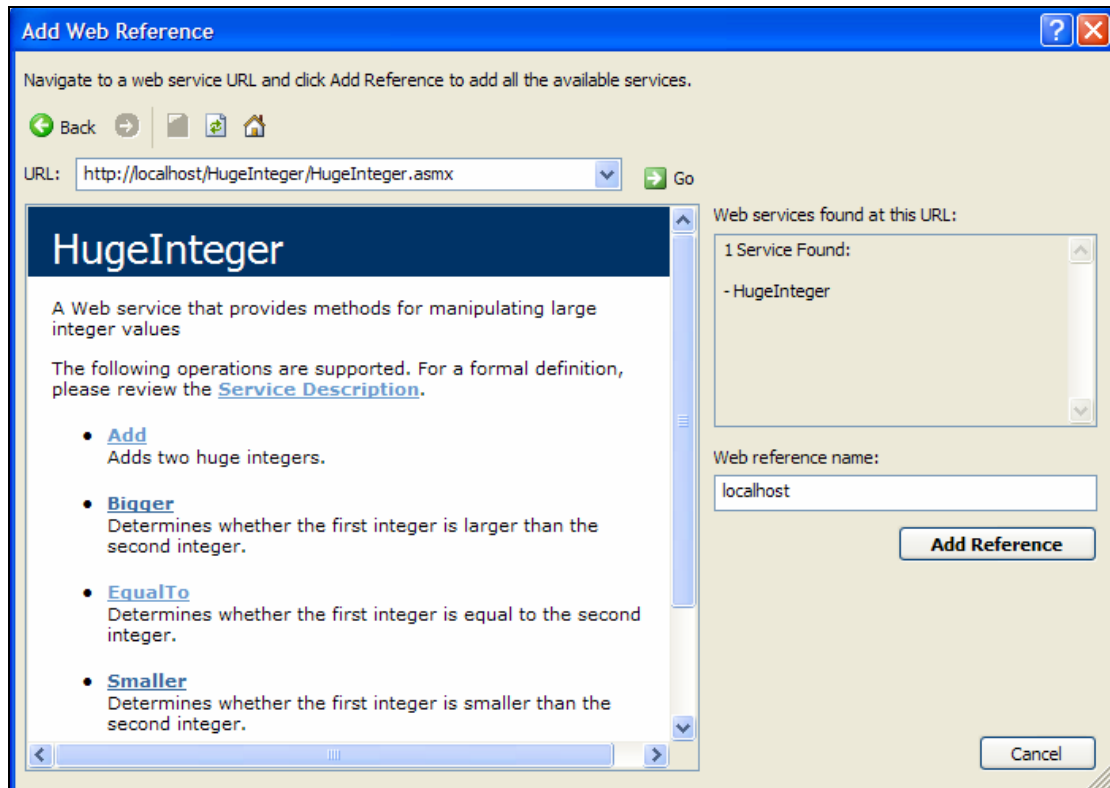
3. الخطوة الثالثة: تحديد الخدمة المطلوبة

اختر الخدمة HugeInteger



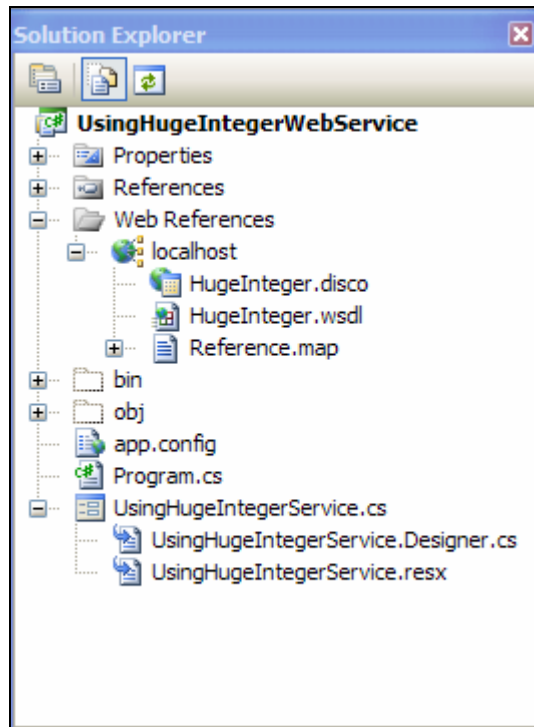
4. الخطوة الرابعة: إضافة مرجع الويب

اختر Add Reference



5. الخطوة الخامسة: معاينة مراجع الويب

افتح الآن المجلد Web References في Solution Explorer وتأكد من وجود ملفات .HugeInteger



استهلاك خدمة ويب

نعرض فيما يلي الكود الذي يستخدم خدمة الويب **HugeInteger** وذلك بهدف القيام بحسابات على أعداد صحيحة تصل إلى 100 خانة.

يصرح السطر 22 عن المتحول **remoteInteger** من النمط **localhost.HugeInteger**. يستخدم هذا المتحول في كل معالجات أحداث التطبيق لاستدعاء طرق **HugeInteger**. يتم إنشاء غرض وكيل **proxy** وإسناده إلى هذا المتحول في السطر 31 من حدث تحميل النموذج **Form_Load**. تقوم الأسطر 52-53، 66-67، 95-96، 116-117، 135-136 باستدعاء طرق خدمة الويب بـ في مختلف معالجات أحداث الأزرار الموضوعة على النموذج:

```
1 // UsingHugeIntegerService.cs
2 // Using the HugeInteger Web Service.
3 using System;
4 using System.Collections.Generic;
5 using System.ComponentModel;
6 using System.Data;
7 using System.Drawing;
8 using System.Text;
9 using System.Windows.Forms;
10 using System.Web.Services.Protocols;
11
12 namespace UsingHugeIntegerWebService
13 {
14     public partial class UsingHugeIntegerServiceForm : Form
15     {
```

```

16 public UsingHugeIntegerServiceForm()
17 {
18     InitializeComponent();
19 } // end constructor
20
21 // declare a reference to Web service
22 private localhost.HugeInteger remoteInteger;
23
24 private char[] zeros={'0'}; //character to trim from strings
25
26 // instantiates object to interact with Web service
27 private void UsingHugeIntegerServiceForm_Load( object sender,
28     EventArgs e )
29 {
30     // instantiate remoteInteger
31     remoteInteger = new localhost.HugeInteger();
32 } // end method UsingHugeIntegerServiceForm_Load
33
34 // adds two numbers input by user
35 private void addButton_Click( object sender, EventArgs e )
36 {
37     // make sure numbers do not exceed 100 digits and that both
38     // are not 100 digits long, which would result in overflow
39     if ( firstTextBox.Text.Length > 100 ||
40         secondTextBox.Text.Length > 100 ||
41         ( firstTextBox.Text.Length == 100 &&
42           secondTextBox.Text.Length == 100 ) )
43     {
44         MessageBox.Show( "HugeIntegers must not be more " +
45             "than 100 digits\r\nBoth integers cannot be " +
46             "of length 100: this causes an overflow", "Error",
47             MessageBoxButtons.OK, MessageBoxIcon.Information );
48         return;
49     } // end if
50
51     // perform addition
52     resultLabel.Text = remoteInteger.Add(
53         firstTextBox.Text, secondTextBox.Text ).TrimStart( zeros );
54 } // end method addButton_Click
55
56 // subtracts two numbers input by user
57 private void subtractButton_Click( object sender, EventArgs e )
58 {
59     // make sure HugeIntegers do not exceed 100 digits
60     if ( SizeCheck( firstTextBox, secondTextBox ) )
61         return;
62
63     // perform subtraction
64     try
65     {
66         string result = remoteInteger.Subtract(
67             firstTextBox.Text, secondTextBox.Text ).TrimStart( zeros );
68
69         if ( result == "" )
70             resultLabel.Text = "0";
71         else
72             resultLabel.Text = result;
73     }

```

```

74     } // end try
75
76     // if WebMethod throws an exception,
77     // then first argument was smaller than second
78     catch ( SoapException exception )
79     {
80         MessageBox.Show(
81             "First argument was smaller than the second" );
82     } // end catch
83 } // end method subtractButton_Click
84
85 // determines whether first number
86 // input by user is larger than second
87 private void largerButton_Click( object sender, EventArgs e )
88 {
89     // make sure HugeIntegers do not exceed 100 digits
90     if ( SizeCheck( firstTextBox, secondTextBox ) )
91         return;
92
93     // call Web-service method to determine if
94     // first integer is larger than the second
95     if ( remoteInteger.Bigger( firstTextBox.Text,
96         secondTextBox.Text ) )
97         resultLabel.Text = firstTextBox.Text.TrimStart( zeros ) +
98             " is larger than " +
99             secondTextBox.Text.TrimStart( zeros );
100     else
101         resultLabel.Text = firstTextBox.Text.TrimStart( zeros ) +
102             " is not larger than " +
103             secondTextBox.Text.TrimStart( zeros );
104     } // end method largerButton_Click
105
106     // determines whether first number
107     // input by user is smaller than second
108 private void smallerButton_Click( object sender, EventArgs e )
109 {
110     // make sure HugeIntegers do not exceed 100 digits
111     if ( SizeCheck( firstTextBox, secondTextBox ) )
112         return;
113
114     // call Web-service method to determine if
115     // first integer is smaller than second
116     if ( remoteInteger.Smaller( firstTextBox.Text,
117         secondTextBox.Text ) )
118         resultLabel.Text = firstTextBox.Text.TrimStart( zeros ) +
119             " is smaller than " +
120             secondTextBox.Text.TrimStart( zeros );
121     else
122         resultLabel.Text = firstTextBox.Text.TrimStart( zeros ) +
123             " is not smaller than " +
124             secondTextBox.Text.TrimStart( zeros );
125     } // end method smallerButton_Click
126
127 // determines whether two numbers input by user are equal
128 private void equalButton_Click( object sender, EventArgs e )
129 {
130     // make sure HugeIntegers do not exceed 100 digits
131     if ( SizeCheck( firstTextBox, secondTextBox ) )

```

```

132         return;
133
134         // call Web-service method to determine if integers are equal
135         if ( remoteInteger.EqualTo( firstTextBox.Text,
136                                     secondTextBox.Text ) )
137             resultLabel.Text = firstTextBox.Text.TrimStart( zeros ) +
138                             " is equal to " + secondTextBox.Text.TrimStart( zeros );
139         else
140             resultLabel.Text = firstTextBox.Text.TrimStart( zeros ) +
141                             " is not equal to " +
142                             secondTextBox.Text.TrimStart( zeros );
143     } // end method equalButton_Click
144
145     // determines whether numbers input by user are too big
146     private bool SizeCheck( TextBox first, TextBox second )
147     {
148         // error message if either number has too many digits
149         if ( ( first.Text.Length > 100 ) ||
150             ( second.Text.Length > 100 ) )
151         {
152             MessageBox.Show("HugeIntegers must be less than 100 digits",
153                             "Error", MessageBoxButtons.OK, MessageBoxIcon.Information);
154             return true;
155         } // end if
156
157         return false;
158     } // end method SizeCheck
159 } // end class UsingHugeIntegerServiceForm
160 } // end namespace UsingHugeIntegerWebService

```

يبين الشكل التالي إضافة عددين:

The screenshot shows a Windows application window titled "Using Huge Integer Service". Inside, there's a text area with the instruction "Please enter two positive numbers up to 100 digits each." Below this are two text boxes. The first text box contains "9999999999" and the second contains "1". Between these text boxes are five buttons: "Add", "Subtract", "Larger Than", "Smaller Tha", and "Equal". The "Add" button is highlighted with a yellow border. Below the buttons is a large text box containing the result "10000000000".

يبين الشكل التالي طرح عددين:

The screenshot shows the same application window. The "Subtract" button is now highlighted with a yellow border. The first text box still contains "9999999999" and the second still contains "1". The result text box at the bottom now shows "9999999998".

يبين الشكل التالي اختبار أكبر من:

Using Huge Integer Service

Please enter two positive numbers up to 100 digits each.

9999999999

1

Add Subtract Larger Than Smaller Tha Equal

9999999999 is larger than 1

يبين الشكل التالي اختبار أصغر من:

Using Huge Integer Service

Please enter two positive numbers up to 100 digits each.

9999999999

1

Add Subtract Larger Than Smaller Tha Equal

9999999999 is not smaller than 1

يبين الشكل التالي اختبار المساواة:

Using Huge Integer Service

Please enter two positive numbers up to 100 digits each.

9999999999

1

Add Subtract Larger Than Smaller Tha Equal

9999999999 is not equal to 1

5. استخدام خدمات الويب في نماذج الويب

استعرضنا في الأمثلة السابقة من هذا الفصل استهلاك خدمات الويب من قبل نمذجة-أدج ويند-دوز Windows Forms. سنرى في هذه الفقرة كيفية استهلاك خدمات الويب من قبل نماذج الويب-Web Forms. لاسيما مع التزايد المطرد لاستخدام نماذج الويب في الأعمال.

سوف نقوم أولاً بإنشاء خدمة ويب تسمح لزيور بإدخال المعلومات عن المقعد الذي يريد حجزه ومن ثم تقوم بحجز المقعد إن كان متاحاً.

ثم سنقوم فيما بعد ببناء تطبيق ويب يستخدم هذه الخدمة من خلال صفحة ويب تـ-سمح بـ-إجراء عملية الحجز.

تملك خدمة الويب ReservationService التالية طريقة واحدة Reserve (الأسطر 24-42) تقوم بالبحث في مفاد قاعدة البيانات Tickets.Mdf لتحديد مقعد يطابق رغبة الزبـ-ون. إذا وجـ-دت الطريقة مقعد تقوم بتعديل قاعدة البيانات لحجزه و تُعيد True. أما إذا لـ-م يـ-تم الحجز ف تُعيد False.

```
1 // ReservationService.cs
2 // Airline reservation Web Service.
3 using System;
4 using System.Web;
5 using System.Web.Services;
6 using System.Web.Services.Protocols;
7
8 [WebService(Namespace = "http://www.svuonline.org/", Description =
9 "Service that enables a user to reserve a seat on a plane." ) ]
10 [ WebServiceBinding( ConformsTo = WsiProfiles.BasicProfile1_1 ) ]
11 public class ReservationService : System.Web.Services.WebService
12 {
13     // create TicketsDataSet object for caching data
14     // from the Tickets database
15     private TicketsDataSet ticketsDataSet = new TicketsDataSet();
16
17     // create SeatsTableAdapter for interacting with the database
18     private TicketsDataSetTableAdapters.SeatsTableAdapter
19     SeatsTableAdapter =
20         new TicketsDataSetTableAdapters.SeatsTableAdapter();
21
22     // checks database to determine whether matching seat is available
23     [ WebMethod( Description = "Method to reserve a seat." ) ]
24     public bool Reserve( string seatType, string classType )
25     {
26         // fill TicketsDataSet.Seats with rows that represent untaken
27         // seats that match the specified seatType and classType
28         SeatsTableAdapter.FillByTypeAndClass(
29             ticketsDataSet.Seats, seatType, classType );
30     }
```

```

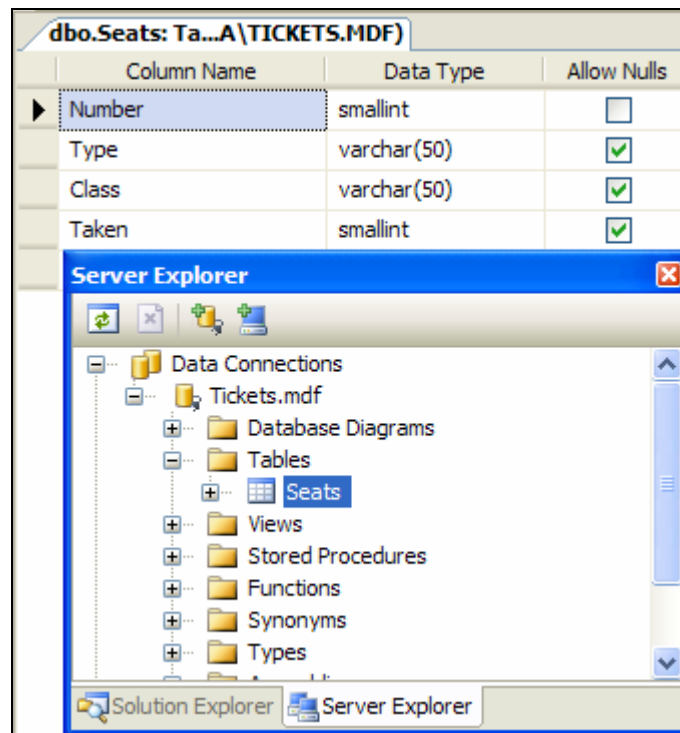
31 // if the number of seats returned is nonzero,
32 // obtain the first matching seat number and mark it as taken
33 if ( ticketsDataSet.Seats.Count != 0 )
34 {
35     string seatNumber = ticketsDataSet.Seats[ 0 ].Number;
36
37     SeatsTableAdapter.UpdateSeatAsTaken( seatNumber );
38     return true; // seat was reserved
39 } // end if
40
41 return false; // no seat was reserved
42 } // end method Reserve
43 } // end class ReservationService

```

تمتلك الطريقة Reserve معاملي دخل، الأول هو سلسلة نصية تُحدد نوع المقعد المطلوب: (Window, Middle, Aisle). والثاني سلسلة نصية تُحدد درجة المقعد: (Economy, First).

يحتوي الجدول Seats في قاعدة البيانات Tickets.mdf أربعة أعمدة:

- Number رقم المقعد.
- Type نوع المقعد (Window, Middle, Aisle).
- Class درجة المقعد (Economy, First).
- Taken محجوز (1) أو متاح (0).



تقوم الأسطر 28-29 بالاستعلام عن أرقام المقاعد المتاحة والتي توافق نوع ودرجة مقعد محددين، تقوم هذه التعليمة بملء الجدول Seat في مجموعة البيانات ticketsDataSet بنتائج الاستعلام التالي:

```
SELECT Number
FROM Seats
WHERE (Taken = 0 ) AND (Type=@type) AND (Class=@class)
```

يتم استبدال معاملات الاستعلام السابق @type و @class بقيمة معملية إدخال الطريقة seatType و classType.

يختبر السطر 33 فيما إذا كان عدد الأسطر في الجدول Seats الناتج عن تنفيذ الاستعلام لا يساوي الصفر (مما يعني وجود مقعد متاح) وفي هذه الحالة تقوم الطريقة بحجز أول مقعد متاح. حيث يتم الحصول على رقم أول مقعد متاح في السطر 35 بالوصول لأول عنصر في الجدول (أي Seats[0]). يقوم بعدها السطر 37 بالاستدعاء الطريقة UpdateSeatsAsTaken للموائم SeatsTableAdapter ممراً لها رقم المقعد والتي تقوم بتنفيذ عبارة SQL التالية:

```
UPDATE Seats
SET Taken=1
WHERE (Number=@number)
```

تعيد الطريقة Reserve القيمة true السطر (38) للدلالة على نجاح الحجز. أما إذا لم يكن هناك مقعداً متاحاً فإنها تُرجع false السطر 41.

إضافة مكونة بيانات إلى خدمة ويب

نستعرض فيما يلي الخطوات الواجب إجرائها في Visual Studio لتهيئة مجموعة بيانات DataSet تسمح لخدمة الويب السابقة بالتفاعل مع قاعدة البيانات Tickets.mdf.

1. الخطوة الأولى: إنشاء خدمة الويب ReservationService وإضافة DataSet إلى المشروع:

- ابدأ بإنشاء مشروع من النمط Web Service وسمّه ReservationService.
- أعد تسمية الملف Service.cs بالاسم ReservationService.cs.
- ضع في الملف ReservationService.cs الكود من الفقرة السابقة.
- أضف مجموعة بيانات DataSet للمشروع وسمّها ticketsDataSet. سينشئ تلقائياً

م- وإلم ج- دول TableAdapter وفي- تج المع- الج Table Adapter Configuration Wizard.

2. الخطوة الثانية: تحديد مصدر البيانات وإنشاء الاتصال

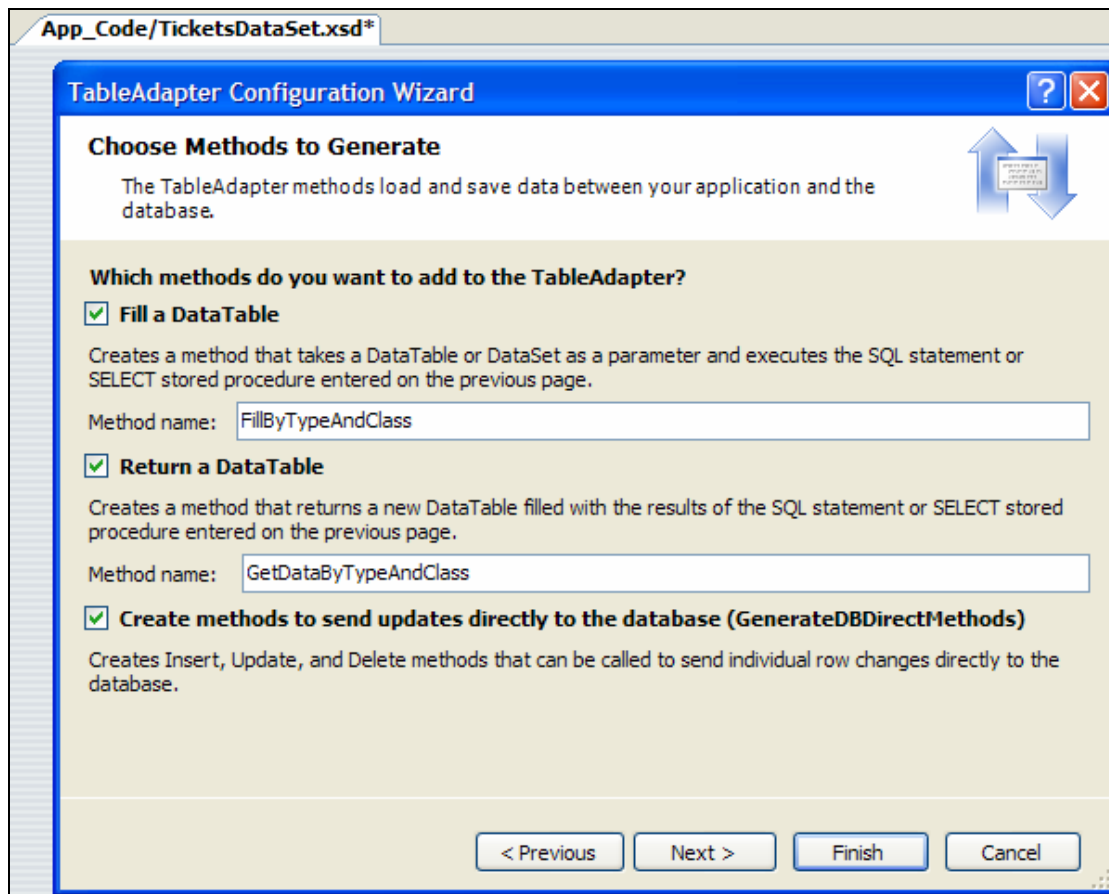
- حدد في نافذة Add Connection م- ن المع- الج الخي- ار Microsoft SQL Server Database File كمصدر البيانات Data Source .
- حدد مسار قاعدة البيانات Tickets.mdf .

3. الخطوة الثالثة: تهيئة استعلام للحصول على المقاعد المتاحة

- حدد الاستعلام التالي كمصدر لبيانات موائم الجدول:

```
SELECT Number  
FROM Seats  
WHERE (Taken = 0 ) AND (Type=@type) AND (Class=@class)
```

أعد تسمية الطرق.



4. الخطوة الرابعة: تهيئة عبارة Update لحجز مقعد

- أضيف استعلام التحديث التالي إلى موائم الجدول:

```
UPDATE Seats
SET Taken=1
WHERE (Number =@number)
```

- أعط الاسم UpdateSeatAsTaken للطريقة الناتجة.

إنشاء نموذج ويب يستهلك خدمة الويب

يبين الشكل التالي تصميم صفحة الويب والتي تسمح للمستخدم بتحديد كل م-ن درجة المقعد (Economy, First) ونوع المقعد (Aisle, Middle, Window).
تستخدم هذه الصفحة خدمة الويب السابقة لتأمين حجز المقعد.

يبين الشكل التالي كود XML للصفحة ReservationClient.aspx السابقة:

```
1 <!-- ReservationClient.aspx -->
2 <!-- Web Form that allows users to reserve seats on a plane. -->
3 <%@ Page Language="C#" AutoEventWireup="true"
4   CodeFile="ReservationClient.aspx.cs"
5   Inherits="ReservationClient" %>
6
7 <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN"
8   "http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">
9
10 <html xmlns="http://www.w3.org/1999/xhtml" >
11   <head id="Head1" runat="server">
12     <title>Ticket Reservation</title>
13   </head>
14
15   <body>
16     <form id="form1" runat="server">
17       <div>
18         <asp:Label ID="instructionsLabel" runat="server"
19           Text="Please select the seat type and class to reserve:">
20         </asp:Label><br /><br />
```

[illegible]

تُعرّف هذه الصفحة قائمتين منسدلتين DropDownList و زر أمر Button .

تُعرّف الأسطر 22-27 قائمة أنواع المقاعد، بينما تُعرّف الأسطر 29-32 قائمة درجة المقعد.

ينفر المستخدم زر الأمر reserveButton (الأسطر 34-36) لإرسال طلبه للمخدم ليقيم بتنفيذ الإجراءات المرتبطة بالزر reserveButton_Click.

تُعرّف الأسطر 38-39 التسمية errorLabel والتي تُستخدم لإظهار الرسائل الموافقة للمستخدم.

يبين الشكل التالي الكود الخلفي ReservationClient.aspx.cs للصفحة السابقة:

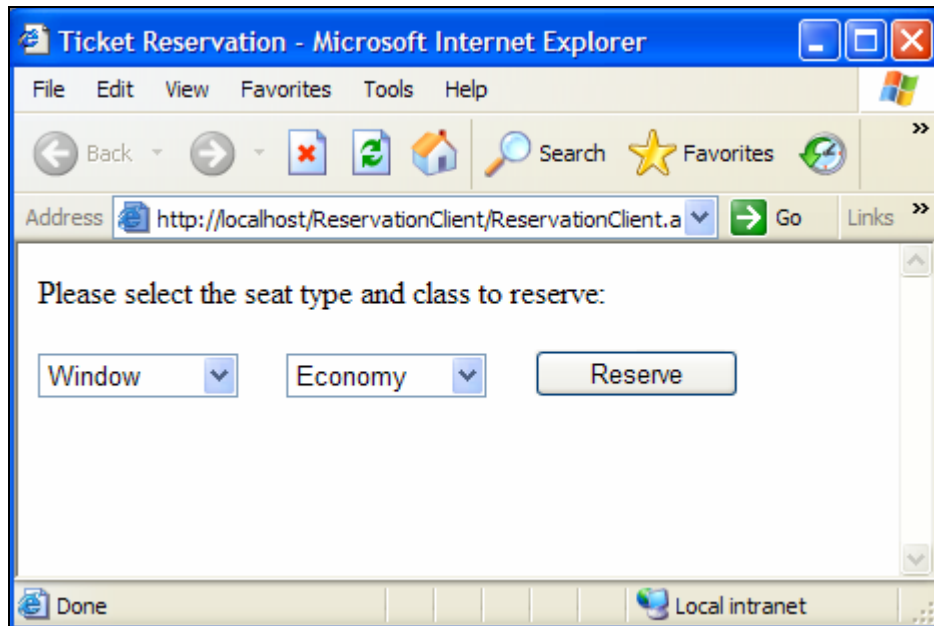
```
1 // ReservationClient.aspx.cs
2 // ReservationClient code behind file.
3 using System;
4 using System.Data;
5 using System.Configuration;
6 using System.Web;
7 using System.Web.Security;
8 using System.Web.UI;
9 using System.Web.UI.WebControls;
10 using System.Web.UI.WebControls.WebParts;
11 using System.Web.UI.HtmlControls;
12
13 public partial class ReservationClient : System.Web.UI.Page
14 {
15     // object of proxy type used to connect to Reservation Web service
16     private localhost.ReservationService ticketAgent =
17         new localhost.ReservationService();
18 }
```

```

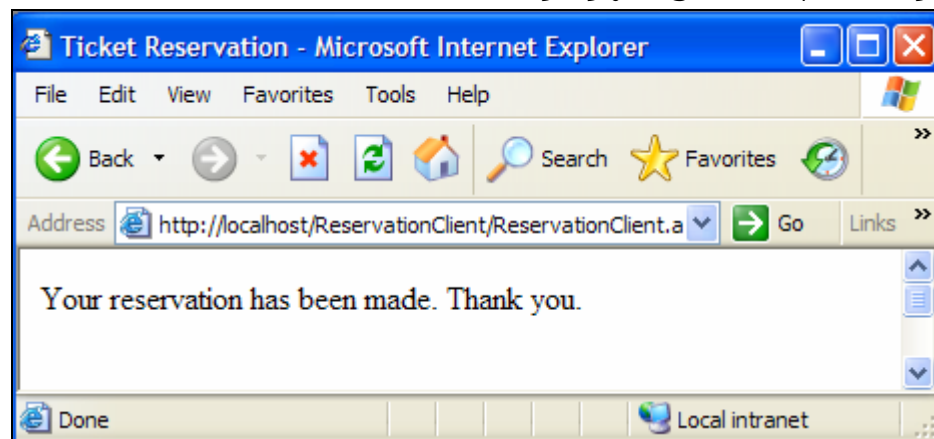
19 // attempt to reserve the selected type of seat
20 protected void reserveButton_Click( object sender, EventArgs e )
21 {
22     // if WebMethod returned true, signal success
23     if ( ticketAgent.Reserve( seatList.SelectedItem.Text,
24         classList.SelectedItem.Text.ToString() ) )
25     {
26         // hide other controls
27         instructionsLabel.Visible = false;
28         seatList.Visible = false;
29         classList.Visible = false;
30         reserveButton.Visible = false;
31         errorLabel.Visible = false;
32
33         // display message indicating success
34         Response.Write( "Your reservation has been made. Thank you." );
35     } // end if
36     else // WebMethod returned false, so signal failure
37     {
38         // display message in the initially blank errorLabel
39         errorLabel.Text = "This type of seat is not available. " +
40             "Please modify your request and try again.";
41     } // end else
42 } // end method reserveButton_Click
43 } // end class ReservationClient

```

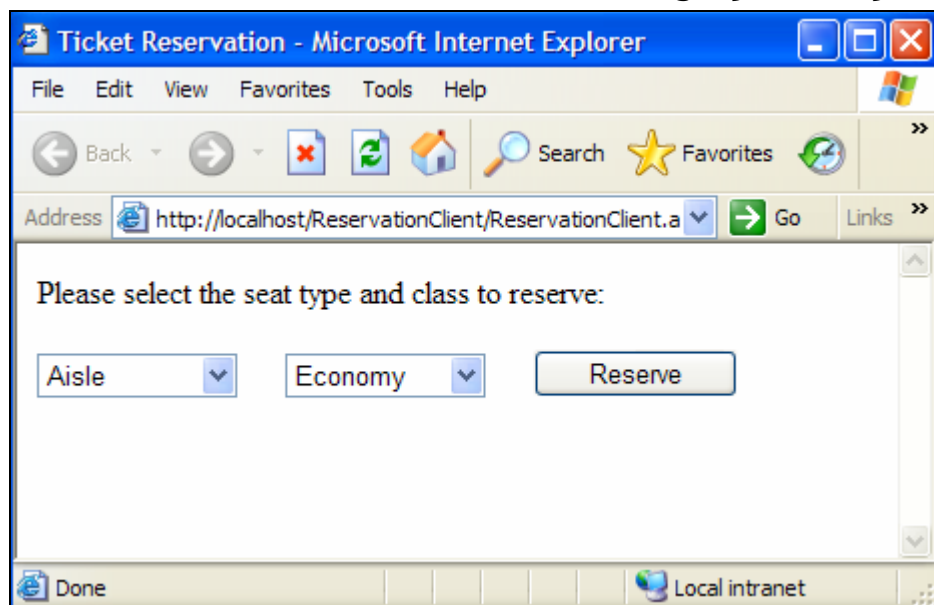
تُنشئ الأسطر (16-17) غرض من النمط **ReservationService**. عندما ينفذ المستخدم على زر **Reserve**، فإن معالج الحدث **reserveButton_Click** الأسطر (20-42) ينفذ. يستدعي معالج الحدث طريقة خدمة الويب **Reserve** ويمرر لها كمعاملتي إدخال نوع المقعد ودرجته الأسطر (23-24). إذا أعادت الطريقة **Reserve** القيمة **True**، تظهر رسالة تُعلم المستخدم بإتمام عملية الحجز. السطر (34). وإلا، توضع رسالة موافقة في التسمية **errorLabel** لإعلام المستخدم بعدم توفر مقعد من المواصفات المحددة. يبين الإيضاح استعراض صفحة الويب.



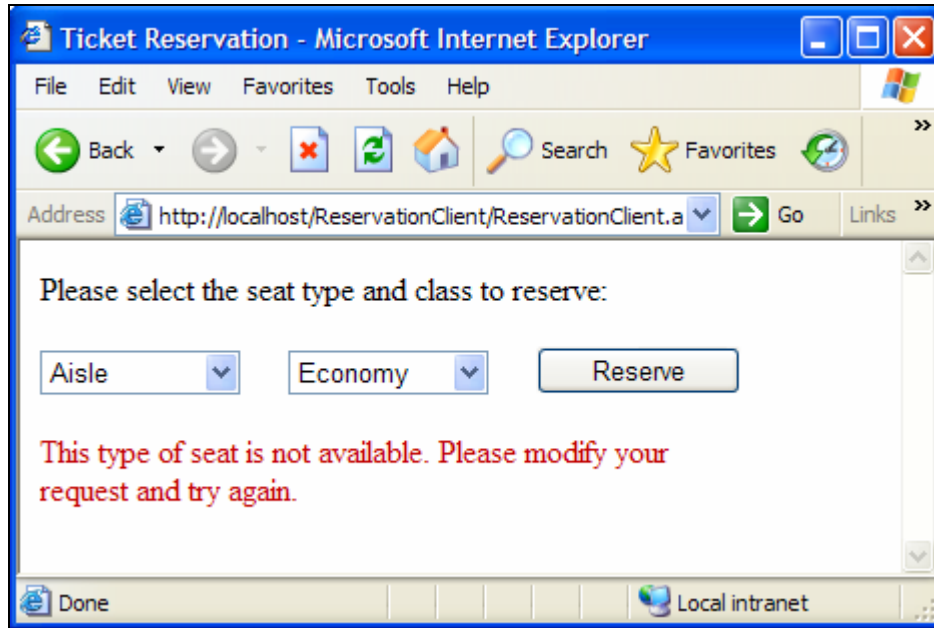
عند اختيار المستخدم لمفعد متاح تظهر الرسالة:



أما عند اختياره لمفعد غير متاح:



فتظهر الرسالة:



6. الأنماط المعرفة في خدمات الويب

تستقبل و تُعيد جميع طرق خدمات الويب التي استعرضناها سابقاً أنماط بيانات بسيطة. يمكن أيضاً معالجة الأنماط المعرفة من قبل المبرمج **User-Defined Types**. حيث يمكن تمرير وإعادة هذه الأنماط من طرق خدمة ويب. كذلك، يمكن للزبائن مستهلكي خدمة الويب - من استخدام هذه الأنماط لأن الصف **proxy** المنشئ يحوي تعريف هذه الأنماط. نعرض في هذه الفقرة خدمة ويب تُدعى **EquationGenerator** تقوم بتوليد معادلات جبرية عشوائية من النمط **Equation**.

يكون الزبون في هذه الحالة برنامج رياضيات تعليمي يقوم بـ سؤال المستخدم عن السؤال المرغوب (جمع، طرح، ضرب) وعن المستوى (أرقام ذات خانة واحدة، خانتين، ثلاث خانات). يمرر هذا البرنامج الزبون رغبات المستخدم لخدمة الويب والتي تُعيد معادلة مع أرقام عشوائية حسب عدد الخانات المطلوب. يستقبل البرنامج الزبون هذه المعادلة ويقوم بعرضها للمستخدم.

وتشمل:

1. النشر المتسلسل للأنماط المعرفة.

2. إنشاء النمط المعروف.

3. إنشاء خدمة الويب.
4. اختبار خدمة الويب.
5. إنشاء الزبون مستهلك خدمة الويب.

النشر المتسلسل للأنماط المعرفة

أشرنا سابقاً بأن جميع الأنماط الممررة من وإلى خدمة ويب يجب أن تُدعم من قبل البروتوكول SOAP. إلا أنه كيف يمكن للبروتوكول SOAP أن يدعم نمط لم ينشئ بعد؟. يتم من أجل ذلك عملية نشر متسلسل للأنماط المعرفة والمرسلة من وإلى خدمة ويب مما يمكن م-ن تمريرهم بتنسيق XML. يدعى هذا الإجراء بالسلسلة XML Serialization.

يجب أن تُحقق الصفوف المستخدمة لتحديد النمط المرجع لطريقة ويب أو لمعاملات طريقة ويب المتطلبات التالية:

1. يجب أن تحوي باني عام public constructor بدون معاملات.
2. يجب أن تكون الخصائص عامة public properties.
3. يجب أن تحوي الخصائص كل من الطريقتين get و set.

إنشاء النمط المعرف

يعرّف الصف التالي النمط Equation. تُعرّف الأسطر 28-46 باني يأخذ ثلاثة مع-املات دخ-ل. رقمين صحيحين يمثلان المعامل اليساري واليمينى، وسلسلة نصية تُمثل العملية الجبرية المطلوبة القيام بها. يقوم هذا الباني بإسناد مع-املات دخ-له إلى حق-ول ال-صف الموافقة leftOperand, rightOperand, resultValue. ومن ثم يقوم بحساب نتيجة المعادلة الجبرية وإسنادها إلى حق-ل الصف resultValue. تُعرّف الأسطر 21-25 باني ليس له مع-املات، ي-وم هذا الباني باستدعاء الباني السابق مع قيم ابتدائية. يعرّف الصف Equation خمسة خصائص:

- LeftHandSide الأسطر (56-68) تُعيد سلسلة نصية تُمثل كل ما يوجد على ي-سار إشارة المساواة.

- RightHandSide الأسطر (71-82) تُعيد سلسلة نصية تُمثل كل ما يوجد على ي-مين إشارة المساواة.

- Left الأسطر (85-96) تُعيد المعامل اليساري (رقم).

- Right الأسطر (99-110) تُعيد المعامل الأيمن (رقم).

- Result الأسطر (114-125) تُعيد الناتج (رقم).

● Operation الأسطر (128-139) تُعيد العملية (سلسلة نصية).

لا يحتاج زبون خدمة الويب إلى تغيير رقم كـ لـ مـ ن الخـ صائـص LeftHandSide و RightHandSide. إلا أنه كما أشرنا سابقاً يجب تعريف طريقتي get و set لكل خاصية.

```
1 // Equation.cs
2 // Class Equation that contains information about an equation.
3 using System;
4 using System.Data;
5 using System.Configuration;
6 using System.Web;
7 using System.Web.Security;
8 using System.Web.UI;
9 using System.Web.UI.WebControls;
10 using System.Web.UI.WebControls.WebParts;
11 using System.Web.UI.HtmlControls;
12
13 public class Equation
14 {
15     private int leftOperand; // number to the left of the operator
16     private int rightOperand; // number to the right of the operator
17     private int resultValue; // result of the operation
18     private string operationType; // type of the operation
19
20     // required default constructor
21     public Equation()
22         : this( 0, 0, "+" )
23     {
24         // empty body
25     } // end default constructor
26
27     // three-argument constructor for class Equation
28     public Equation( int leftValue, int rightValue, string type )
29     {
30         leftOperand = leftValue;
31         rightOperand = rightValue;
32         operationType = type;
33
34         switch ( operationType ) // perform appropriate operation
35         {
36             case "+": // addition
37                 resultValue = leftOperand + rightOperand;
38                 break;
39             case "-": // subtraction
40                 resultValue = leftOperand - rightOperand;
41                 break;
42             case "*": // multiplication
43                 resultValue = leftOperand * rightOperand;
44                 break;
45         } // end switch
46     } // end three-argument constructor
47
48     // return string representation of the Equation object
49     public override string ToString()
50     {
```

```

51     return leftOperand.ToString() + " " + operationType + " " +
52         rightOperand.ToString() + " = " + resultValue.ToString();
53 } // end method ToString
54
55 // property that returns a string representing left-hand side
56 public string LeftHandSide
57 {
58     get
59     {
60         return leftOperand.ToString() + " " + operationType + " " +
61             rightOperand.ToString();
62     } // end get
63
64     set // required set accessor
65     {
66         // empty body
67     } // end set
68 } // end property LeftHandSide
69
70 // property that returns a string representing right-hand side
71 public string RightHandSide
72 {
73     get
74     {
75         return resultValue.ToString();
76     } // end get
77
78     set // required set accessor
79     {
80         // empty body
81     } // end set
82 } // end property RightHandSide
83
84 // property to access the left operand
85 public int Left
86 {
87     get
88 {
89         return leftOperand;
90     } // end get
91
92     set
93     {
94         leftOperand = value;
95     } // end set
96 } // end property Left
97
98 // property to access the right operand
99 public int Right
100 {
101     get
102     {
103         return rightOperand;
104     } // end get
105
106     set
107     {
108         rightOperand = value;
109     } // end set

```

```

110 } // end property Right
111
112 // property to access the result of applying
113 // an operation to the left and right operands
114 public int Result
115 {
116     get
117     {
118         return resultValue;
119     } // end get
120
121     set
122     {
123         resultValue = value;
124     } // end set
125 } // end property Result
126
127 // property to access the operation
128 public string Operation
129 {
130     get
131     {
132         return operationType;
133     } // end get
134
135     set
136     {
137         operationType = value;
138     } // end set
139 } // end property Operation
140 } // end class Equation

```

إنشاء خدمة الويب

تقوم خدمة الويب التالية **EquationGenerator** بإنشاء معادلات جبرية عشوائية. تحـوي هـذه الخدمة فقط طريقة الويب **GenerateEquation** الأسطر (16-32) والتي تأخذ معـاملين الأول سلسلة نصية تُمثل العملية الجبرية والثاني رقم يمثل صعوبة المعادلة حيث يأخذ القيم (1، 2، 3) والتي تكون عدد خانات المعاملات.

```

1 // Generator.cs
2 // Web Service to generate random equations based on a specified
3 // operation and difficulty level.
4 using System;
5 using System.Web;
6 using System.Web.Services;
7 using System.Web.Services.Protocols;
8
9 [ WebService( Namespace = "http://www.deitel.com/", Description =
10     "Web service that generates a math equation." ) ]
11 [ WebServiceBinding( ConformsTo = WsiProfiles.BasicProfile1_1 ) ]
12 public class Generator : System.Web.Services.WebService
13 {

```

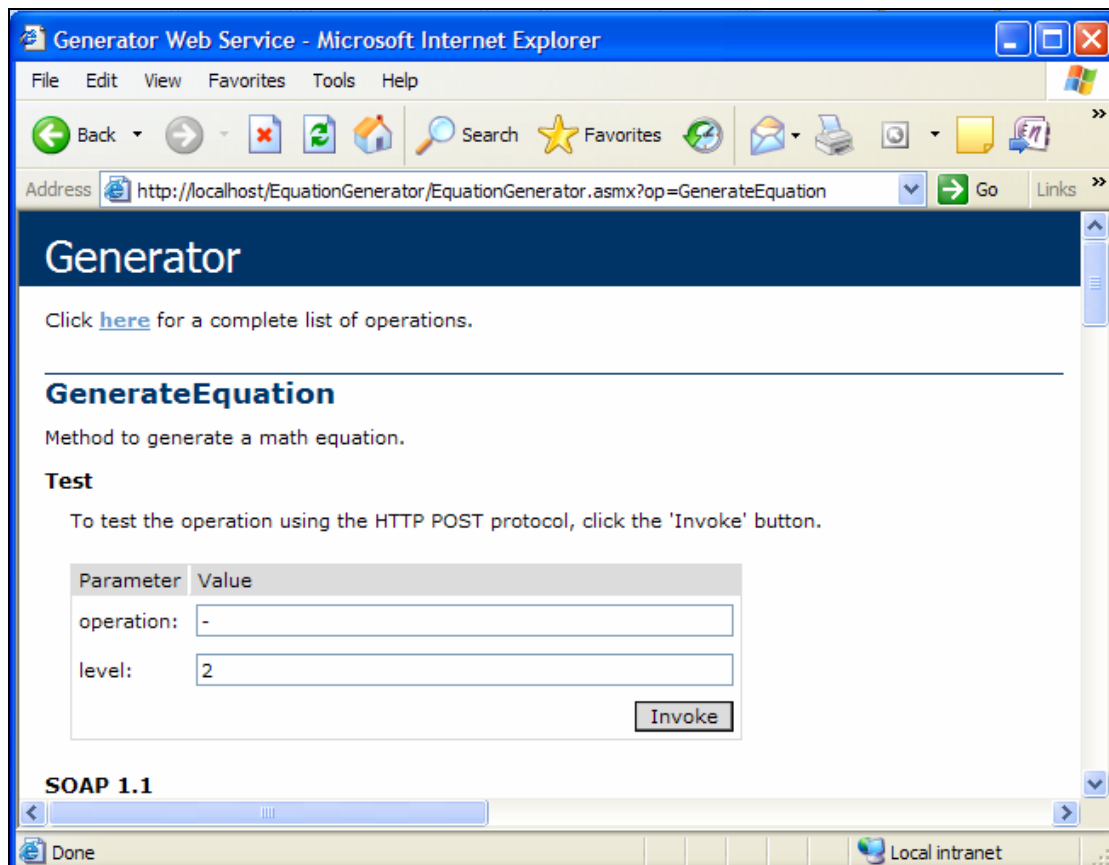
```

14 // Method to generate a math equation
15 [WebMethod(Description = "Method to generate a math equation." ) ]
16 public Equation GenerateEquation( string operation, int level )
17 {
18     // find maximum and minimum number to be used
19     int maximum = Convert.ToInt32( Math.Pow( 10, level ) );
20     int minimum = Convert.ToInt32( Math.Pow( 10, level - 1 ) );
21
22     // object to generate random numbers
23     Random randomObject = new Random();
24
25     // create equation consisting of two random
26     // numbers between minimum and maximum parameters
27     Equation equation = new Equation(
28         randomObject.Next( minimum, maximum ),
29         randomObject.Next( minimum, maximum ), operation );
30
31     return equation;
32 } // end method GenerateEquation
33 } // end class Generator

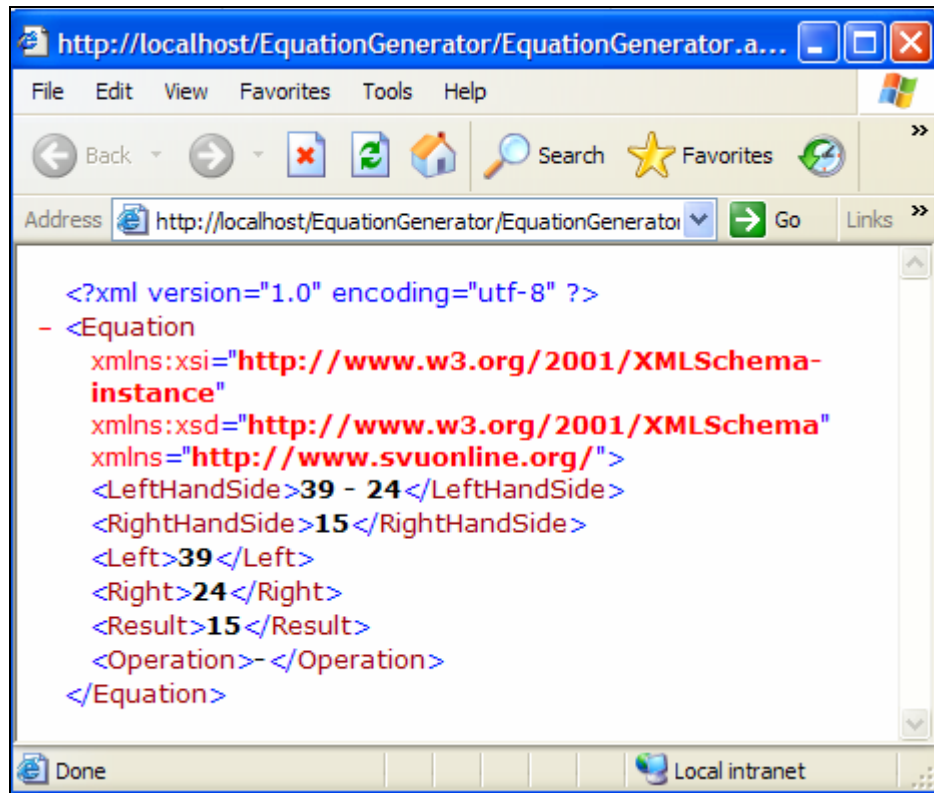
```

اختبار خدمة الويب

يبين الشكل التالي استدعاء طريقة الويب **EquationGenerator**:



كما يبين الشكل التالي ناتج الاستدعاء والذي هو عبارة عن ملف XML. يحدد هذا الملف جميع قيم الخصائص العامة **public** وبيانات الغرض المرجع



إنشاء الزبون مستهلك خدمة الويب

يستخدم التطبيق **MathTutor** التالي خدمة الويب **EquationGenerator**. يستدعي هذا التطبيق طريقة خدمة الويب **GenerateEquation** لإنشاء غرض من النمط **Equation**. ثم يقوم بعرض الطرف الأيسر من المعادلة وينتظر جواب المستخدم.

```
1 // MathTutor.cs
2 //Math tutoring program using WS to generate random equations.
3 using System;
4 using System.Collections.Generic;
5 using System.ComponentModel;
6 using System.Data;
7 using System.Drawing;
8 using System.Text;
9 using System.Windows.Forms;
10
11 namespace MathTutor
12 {
13     public partial class MathTutorForm : Form
14     {
15         public MathTutorForm()
16         {
```

```

17     InitializeComponent();
18 } // end constructor
19
20 private string operation = "+";
21 private int level = 1;
22 private localhost.Equation equation;
23 private localhost.Generator generator = new localhost.Generator();
24
25 // generates new equation when user clicks button
26 private void generateButton_Click( object sender, EventArgs e )
27 {
28     // generate equation using current operation and level
29     equation = generator.GenerateEquation( operation, level );
30
31     // display left-hand side of equation
32     questionLabel.Text = equation.LeftHandSide;
33
34     okButton.Enabled = true;
35     answerTextBox.Enabled = true;
36 } // end method generateButton_Click
37
38 // check user's answer
39 private void okButton_Click( object sender, EventArgs e )
40 {
41     // determine correct result from Equation object
42     int answer = equation.Result;
43
44     if ( answerTextBox.Text == "" )
45         return;
46
47     // get user's answer
48     int userAnswer = Int32.Parse( answerTextBox.Text );
49
50     // determine whether user's answer is correct
51     if ( answer == userAnswer )
52     {
53         questionLabel.Text = ""; // clear question
54         answerTextBox.Text = ""; // clear answer
55         okButton.Enabled = false; // disable OK button
56         MessageBox.Show( "Correct! Good job!" );
57     } // end if
58     else
59         MessageBox.Show( "Incorrect. Try again." );
60 } // end method okButton_Click
61
62 // set difficulty level to 1
63 private void levelOneRadioButton_CheckedChanged( object sender,
64     EventArgs e )
65 {
66     level = 1;
67 } // end method levelOneRadioButton_CheckedChanged
68
69 // set difficulty level to 2
70 private void levelTwoRadioButton_CheckedChanged( object sender,
71     EventArgs e )
72 {
73     level = 2;
74 } // end method levelTwoRadioButton_CheckedChanged
75

```

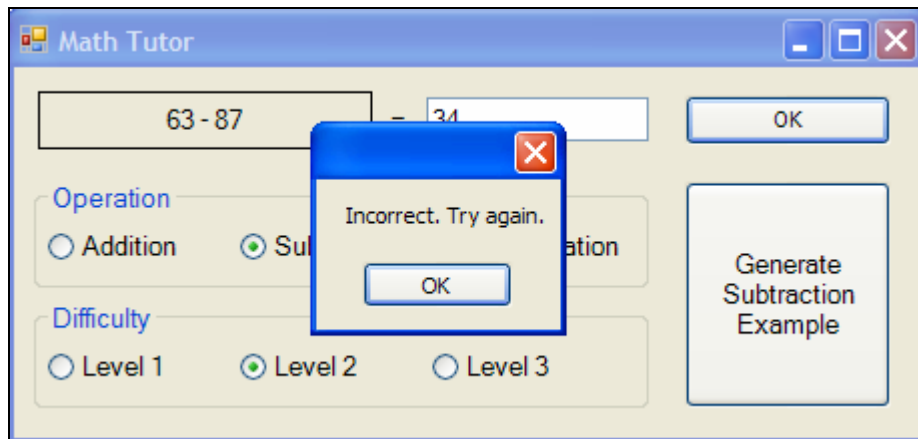
```

76 // set difficulty level to 3
77 private void levelThreeRadioButton_CheckedChanged( object sender,
78     EventArgs e )
79 {
80     level = 3;
81 } // end method levelThreeRadioButton_CheckedChanged
82
83 // set the operation to addition
84 private void additionRadioButton_CheckedChanged( object sender,
85     EventArgs e )
86 {
87     operation = "+";
88     generateButton.Text =
89         "Generate " + additionRadioButton.Text + " Example";
90 } // end method additionRadioButton_CheckedChanged
91
92 // set the operation to subtraction
93 private void subtractionRadioButton_CheckedChanged( object sender,
94     EventArgs e )
95 {
96     operation = "-";
97     generateButton.Text = "Generate " +
98         subtractionRadioButton.Text + " Example";
99 } // end method subtractionRadioButton_CheckedChanged
100
101 // set the operation to multiplication
102 private void multiplicationRadioButton_CheckedChanged(
103     object sender, EventArgs e )
104 {
105     operation = "*";
106     generateButton.Text = "Generate " +
107         multiplicationRadioButton.Text + " Example";
108 } // end method multiplicationRadioButton_CheckedChanged
109 } // end class MathTutorForm
110 } // end namespace MathTutor

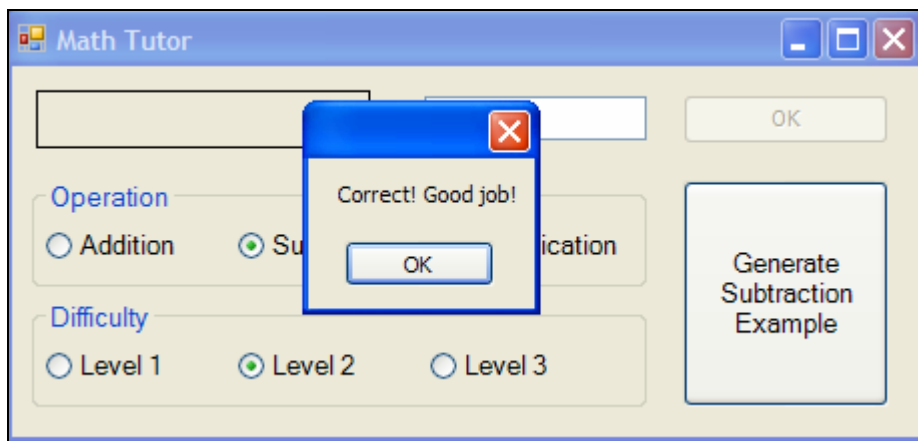
```

يبين الشكل التالي تنفيذ البرنامج:

في حال إدخال قيمة خاطئة تظهر رسالة خطأ موافقة:



وفي حال إدخال قيمة الجواب الصحيح:



7. الخلاصة

استعرضنا في هذا الفصل خدمات الويب كتقانة تسمح للمبرمجين بطلب واستقبال نتائج تنفيذ طرق عبر الشبكة مما يسمح باستخدام أمثلي للطرق في النظم الموزعة (إعادة استخدام، توزيع المهام، ...).

كما بينا بأن خدمة الويب هي صنف يسمح لمختلف الزبائن باستدعاء طرقه عبر الشبكة باستخدام قوالب وبروتوكولات مثل XML و HTTP و SOAP.

أوضحنا الميزات العديدة لهذا المنهج في الحسابات الموزعة، كوصول الزبائن إلى بيانات محددة على حواسيب بعيدة وتمتعهم بمفدرات هذه الحواسيب البعيدة.