http://www.johanyak.hu
e-mail: johanyak.csaba@gamf.kefo.hu

XML Webszolgáltatás alapú osztott alkalmazás fejlesztése

Johanyák Zsolt Csaba[1]

A gyakorlat célja a webszolgáltatások létrehozásának és igénybe vételének elsajátítása egyszerű példákon keresztül.

1. Két szám összegét kiszámoló webszolgáltatás készítése és igénybe vétele

A webszolgáltatás egy ASP.NET-ben megírt szerver oldali alkalmazás, amit a HTTP és SOAP protokollok segítségével hívhatunk meg. Gyakorlatilag egy távoli eljáráshívásra kerül sor, a webszerveren levő objektum egy metódusát fogjuk igénybe venni egy kliens alkalmazásból. Elsőként a szerver alkalmazást készítjük el, majd ezt követően egy Windows Forms kliens alkalmazást fejlesztünk.

1.1. Szerver oldali alkalmazás (webszolgáltatás)

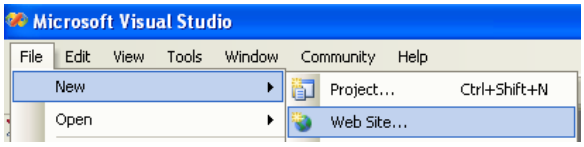
A feladat végrehajtása során a következő fontosabb lépésekre kerül sor:

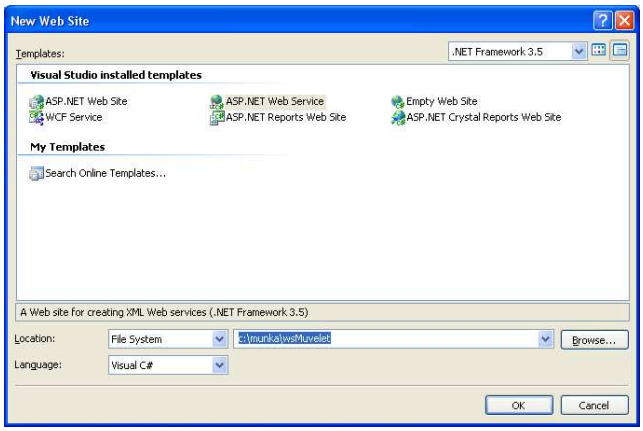
1. Projekt létrehozása
2. Állományok és osztály átnevezése
3. Névtér azonosító megadása
4. Automatikusan generált webszolgáltatás kipróbálása
5. Webszolgáltatásként működő saját metódus definiálása és tesztelése

1.1.1. Projekt létrehozása

Hozzunk létre egy új C# projektet (File/New/Web Site…), melynek típusa (Templates) ASP.NET Web Service és tárolási helye a lokális állományrendszer (File System) c:\munka könyvtára. A projekt neve legyen **wsMuvelet**.

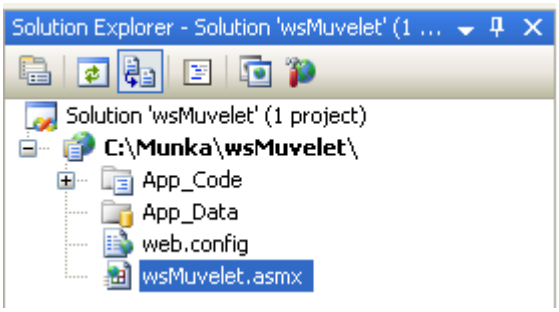
A Visual Studio rendelkezik egy beépített egyszerű webszerverrel, aminek az a célja, hogy segítségével kipróbáljuk a fejlesztett alkalmazásokat. Használatának előfeltétele az, hogy a projekt helyének File System-et adjuk meg.





[1] http://www.johanyak.hu
e-mail:johanyak.csaba@gamf.kefo.hu

1.1.2. Állományok és osztály átnevezése



Nevezzük át a Service.asmx állományt a Solution Explorerben wsMuvelet.asmx-re, majd nyissuk meg kódnézetben, ha nem nyílik meg automatikusan.
Solution Explorerben nyissuk meg a App_Code mappát. A benne található Service.cs állományt nevezzük át wsMuvelet.cs-re. A kódszerkesztőben a wsMuvelet.cs állományban a szolgáltatás osztályát

```
12  public class wsMűvelet : System.Web.Services.WebService
13  {
14      public Service (
15
16          //Uncomment the following line if using designed components
17          //InitializeComponent();
18      }
```

Rename 'Service' to 'wsMűvelet'
Rename with preview...

(Service) és annak konstruktorát nevezzük át wsMűvelet-re.
A wsMuvelet.asmx állományban a hivatkozásokat is módosítsuk a fentieknek megfelelően az alábbi ábra szerint.

1.1.3. Névtér azonosító megadása

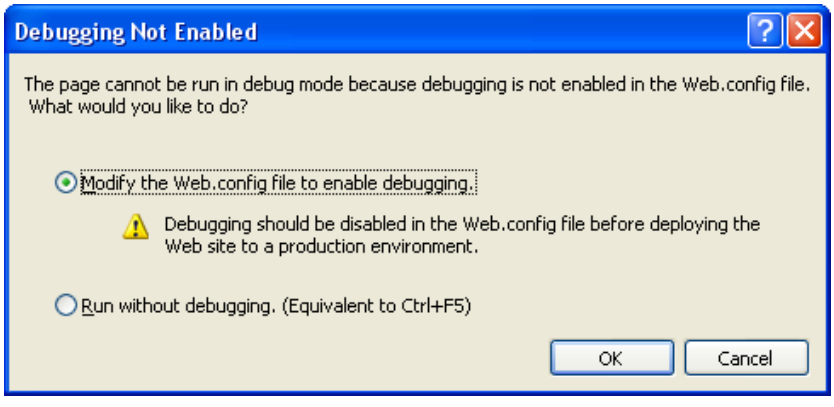
A wsMűvelet osztály definíciója előtt (wsMuvelet.cs) megjelenő attribútumok teszik lehetővé az osztály számára, hogy webszolgáltatásként működő metódusokat definiáljon. A WebService attribútum paramétereként egy egyedi névtér azonosítót szokás megadni. Ennek érdekében írjuk át értékét az alábbiak szerint:

[WebService(Namespace="http://VezeteknévKeresztnév/wsMűvelet/")]

A névtér-azonosítóként konvencionálisan URI formátumot használunk, ez természetesen nem valódi URL. A **VezeteknévKeresztnév** részben mindenki a saját nevét adja meg.

1.1.3. Automatikusan generált webszolgáltatás kipróbálása

A Visual Studio Automatikusan generál az osztállyal együtt egy HelloWorld nevű metódust. Az előtte elhelyezett `[WebMethod]` attribútum teszi lehetővé, hogy a metódus webszolgáltatást



nyújthasson. Kipróbálásához válasszuk ki a Debug/Start Debugging menüpontot. Ekkor egy figyelmeztető ablak jelenik meg, amelyben engedélyezhetjük a Debug üzemmódban történő futtatást. Ha ezt tesszük, akkor a Visual Studio létrehoz egy Web.config nevű állományt (ld. alább) a projektben, amelyben beállítja a Debug mód engedélyezését. Ezt csak a fejlesztés során engedélyezzük! Értékét a végső változatban állítsuk false-ra.
A beépített webszerver indulását a tálcán megjelenő kis ikon és magyarázat buborék jelzi.

```
<!--
        Set compilation debug="true" to insert debugging
        symbols into the compiled page. Because this
        affects performance, set this value to true only
        during development.
    -->
<compilation debug="true"/>
```

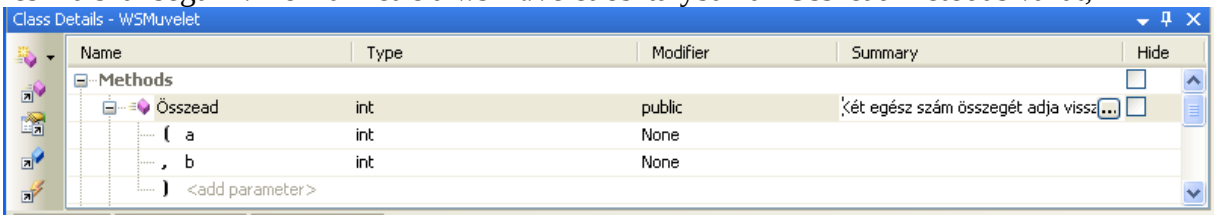
Ekkor egy web böngésző ablak nyílik meg a jobb oldalt látható tartalommal. A HelloWorld hivatkozásra kattintva egy újabb weblap nyílik meg, amelyen megtekinthetjük, hogy milyen formában küldi a kliens a szerverhez a metódushívásokat, illetve milyen formában érkeznek a válaszok (futás eredménye, visszatérési érték). A két folyamat közötti üzenetváltás SOAP protokoll segítségével történik XML formátumban. Ugyancsak ezen a weblapon jelenik meg egy indítás feliratú nyomógomb, aminek segítségével tesztelhetjük az automatikusan generált webszolgáltatást. A tesztelés


```
<?xml version="1.0" encoding="utf-8" ?>
<string
  xmlns="http://JohanyákCsaba/WSMuvelet/">Hello
  World</string>
```

eredménye szintén a böngészőablakban jelenik meg. A böngészőablakban megjelenített tartalom kismértékben eltérő lehet attól függően, hogy melyik böngésző lett alapértelmezettként beállítva.
Fontosnak tartom megjegyezni, hogy a webszolgáltatásokat normál használat során nem web böngészőből vesszük igénybe, hanem egy kliens alkalmazásból (a jelen gyakorlatban ez egy Windows Forms alkalmazás lesz) valamely objektum egy metódusaként fogjuk őket meghívni.

1.1.4. Webszolgáltatásként működő saját metódus definiálása és tesztelése

Egy saját metódust szeretnénk készíteni, ami két egész számot vesz át, és visszaadja az összegüket. Lépjünk ki Debug módból. Készítsük el a wsMuvelet projekt osztálydiagramját. Tegyük megjegyzésbe az automatikusan generált HelloWorld metódust, a továbbiakban nem lesz rá szükségünk. Hozzuk létre a wsMűvelet osztályban az Összead metódus vázát,



majd a kódszerkesztőben írjuk meg a számítási utasítást:

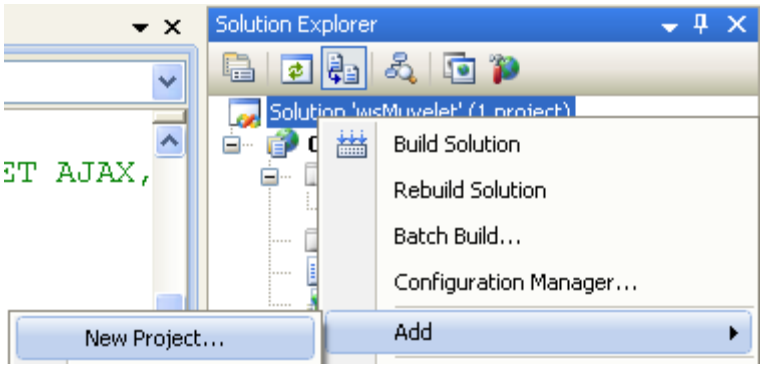
```
/// <summary>
/// Két egész szám összegét adja vissza
/// </summary>
[WebMethod]
public int Összead(int a, int b)
{
  return a+b;
}
```

Ne feledkezzünk meg a metódus fejléce előtt elhelyezendő [WebMethod] attribútumról! Az előzőekben látott módon próbáljuk ki a metódust. A tesztelés során most egy űrlapot kapunk amelynek segítségével megadhatjuk a metódus két paraméterét. Az eredményt az előzőekhez hasonlóan kapjuk meg.


```
<?xml version="1.0" encoding="utf-8" ?>
<int
  xmlns="http://JohanyákCsaba/WSMuvelet/">3</int>
```

A webszolgáltatás üzemszerű használata során egy webszerveren helyezkedik el. Az ehhez kapcsolódó beállítások és egy hosszabb lélegzetű példa alkalmazás a **MandelbrotWebszolgáltatás**, amit egyéni munka keretében kell áttanulmányozni.

1.2. Kliens alkalmazás létrehozása

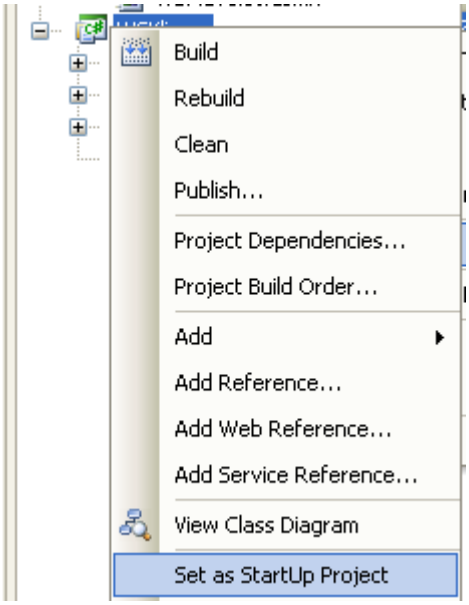


A következőekben egy olyan Windows Forms alkalmazást készítünk, ami igénybe veszi az előzőekben elkészített webszolgáltatást. A fejlesztés során nem egy távoli webszervert fogunk igénybe venni, hanem lokálisan a megoldáson belül különálló projektként (wsMuvelet) jelenlevő webszolgáltatást használjuk.
A feladat végrehajtása során a következő fontosabb lépésekre kerül sor:

1. Új projekt létrehozása és beállítása alapértelmezettként
2. Kapcsolat a webszolgáltatáshoz, proxy osztály létrehozása
3. Kliens alkalmazás felületének kialakítása

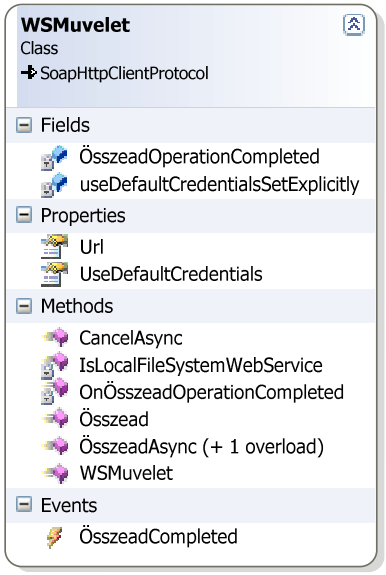
1.2.1. Új projekt létrehozása

A legegyszerűbb megoldás az, ha a korábbi projekt bezárása nélkül hozunk létre egy új projektet. Például Solution Explorerben jobb egérgomb kattintás a megoldás nevén, a gyorsmenüben Add/New Project…. A projekt típusa C# Windows Application, neve wsKliens, helye C:\munka.
Állítsuk be az új projektet alapértelmezettként annak érdekében, hogy egyszerűen és gyorsan indítható legyen. Ehhez Solution Explorerben jobb egérgombbal kattintunk a kliens projekt nevén, majd a gyorsmenüben Set as StartUp Project. Ekkor félkövéren jelenik meg a projekt neve.



1.2.2. Kapcsolat a webszolgáltatáshoz, proxy osztály létrehozása

A webszolgáltatás igénybe vételéhez a projektünknek információkkal kell rendelkeznie a webszolgáltatásról. Ehhez Solution Explorerben jobb egérgombbal kattintunk a kliens projekt nevén, majd a gyorsmenüben Add Service Reference… a megjelenő párbeszédablakban a Discover listát legördítve a Services in solution-t választjuk. A webszolgáltatások listájában a WSMuveletre kattintunk, majd kis várakozás után jobb oldalon a Web reference name mezőben az eredeti localhost kitörlése után megadjuk a WSMuvelet hivatkozási nevet, és kattintunk az Add Reference gombon.
Ekkor a Solution Explorerben a projekt mappáján belül egy újabb mappa kell megjelenjen Web References néven. A mappában egyetlen elem szerepel, a WSMuvelet.
Készítsük el a WSKliens projekt osztálydiagramját. Egyebek között keletkezett egy osztály WSMuvelet néven, ami segítséget nyújt a webszolgáltatás eléréséhez.
Ez egy ún. proxy vagy más néven csonk (stub) osztály, ami elfedi azt a tényt, hogy az összeadást egy másik folyamat egy metódusa fogja végrehajtani. A kliens alkalmazásunkban ebből az osztályból fogunk egy objektumot létrehozni, és ennek az objektumnak fogjuk meghívni az Összeadás metódusát, amikor a művelet végrehajtására lesz szükségünk. A WSMuvelet osztály gondoskodik arról, hogy a beérkező metódushívásokból SOAP üzeneteket generáljon, azokat elküldje a szerverre, fogadja a szerver válaszát, és biztosítsa a visszatérési értéket. Az osztály URL tulajdonsága segítségével állíthatjuk be a telepített webszolgáltatás pontos elérési címét, vagy módosíthatjuk azt akár futásidőben. Az osztály biztosítja az aszinkron hívás lehetőségét is.



1.2.3. Kliens alkalmazás felületének kialakítása

A kliens alkalmazás felületét a mellékelt ábrának megfelelően alakítsuk ki. Alkalmazzuk a már megszokott elnevezési konvenciót. A főablak állományneve legyen frmFoablak, osztályneve legyen ugyanennek ékezetes változata. A szerkesztőmezők nevei: tba, tbb, tbab. A tbab csak olvasható. A nyomógomb neve btPlusz.

1.2.4. Összeadási művelet

Hozzunk létre egy adattagot az frmFőablak osztályban, aminek az lesz a feladata, hogy a webszolgáltatást megtestesítő lokális objektum referenciáját tárolja.

```
private WSKliens.WSMuvelet.WSMuvelet Szerver;
```

A főablak konstruktorában hozzunk létre egy objektumot ehhez az adattaghoz.

```
public frmFőablak()
 {
   InitializeComponent();
   Szerver = new WSKliens.WSMuvelet.WSMuvelet();
 }
```

Készítsünk egy eseménykezelőt a nyomógomb Click eseményéhez. Ebben ellenőrzötten beolvassunk a felhasználó által megadott két számot, meghívjuk a webszolgáltatást, és az eredményt megjelenítjük a tbab szövegmezőben.

```
private void btPlusz_Click(object sender, EventArgs e)
{
  try
  {
    int a = int.Parse(tba.Text);
    int b = int.Parse(tbb.Text);
    int ab = Szerver.Összead(a, b);
    tbab.Text = ab.ToString();
  }
  catch (FormatException)
  {
    MessageBox.Show("A megadott adatok hibásak!");
  }
  catch (System.Net.WebException we)
  {
    MessageBox.Show(we.Message);
  }
}
```

A webszolgáltatás meghívása alatt WebException kivétel keletkezhet. Ilyenkor megjelenítjük a kivétel objektummal kapott rendszer-hibaüzenetet.
Próbáljuk ki a programot.

2. Más által készített webszolgáltatás igénybe vétele

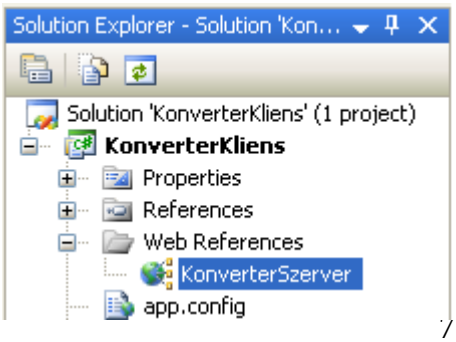
A **http://konverter.weblapportal.hu/Konverter.asmx** címen egy webszolgáltatást helyeztem el, ami két metódust tartalmaz Celsius és Fahrenheit fokok közötti átváltásra. Az alábbiakban egy kliens programot készítünk, ami ezen szolgáltatás segítségével elvégzi a kívánt átalakítást.

2.1. Új projekt létrehozása

A feladat megoldásához hozzunk létre egy Windows Application típusú C# projektet KonverterKliens néven.

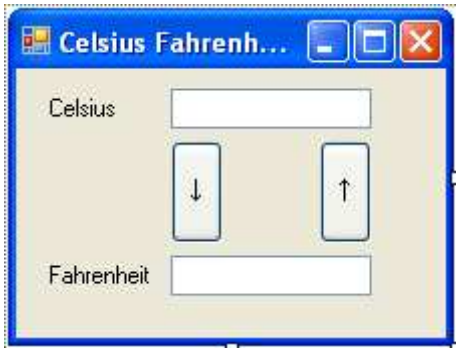
2.2. Kapcsolat a webszolgáltatáshoz, proxy osztály létrehozása

Solution Explorerben jobb egérgombbal kattintunk a kliens projekt nevén, majd a gyorsmenüben Add Web Reference… a megjelenő párbeszédablak URL mezőjében megadjuk a **http://konverter.weblapportal.hu/Konverter.asmx** címet, majd kattintunk a Go gombon. Ezután jobb oldalon a Web reference name mezőben megadjuk a



KonverterSzerver hivatkozási nevet, és kattintunk az Add Reference gombon. Ekkor a Solution Explorerben a projekt mappáján belül egy újabb mappa kell megjelenjen Web References néven. A mappában egyetlen elem szerepel, a KonverterSzerver.

2.3. Kliens alkalmazás felületének kialakítása



A kliens alkalmazás felületét a mellékelt ábrának megfelelően alakítsuk ki. Alkalmazzuk a már megszokott elnevezési konvenciót. A főablak állományneve legyen frmFoablak, osztályneve legyen ugyanennek ékezetes változata. A szerkesztőmezők nevei: tbCelsius és tbFahrenheit. A nyomógombok neve btC2F és btF2C. A nyomógombokon szereplő nyilakat Symbol betűtípussal tudjuk megjeleníteni (Font tulajdonság).

2.4. Webszolgáltatás igénybe vétele

Hozzunk létre egy adattagot az frmFőablak osztályban, aminek az lesz a feladata, hogy a webszolgáltatást megtestesítő lokális objektum referenciáját tárolja.

```
private KonverterSzerver.Konverter Konverter;
```

A főablak konstruktorában hozzunk létre egy objektumot ehhez az adattaghoz.

```
public frmFőablak()
{
  InitializeComponent();
  Konverter=new KonverterSzerver.Konverter();
}
```

Valósítsuk meg a Celsius→Fahrenheit irányú átalakítást. Ennek érdekében készítsünk a bal oldali nyomógombhoz egy eseménykezelőt, amiben beolvassunk a Celsius szövegmezőben elhelyezett számadatot, átalakítjuk számmá, majd meghívjuk a webszolgáltatás megfelelő metódusát.

```
private void btC2F_Click(object sender, EventArgs e)
{ try
  { double Celsius = double.Parse(tbCelsius.Text);
    double Fahrenheit =
      Konverter.Celsius2Fahrenheit(Celsius);
    tbFahrenheit.Text = Fahrenheit.ToString();
  }
  catch (FormatException)
  { MessageBox.Show("A megadott adat hibás!");
  }
  catch (System.Net.WebException we)
  { MessageBox.Show(we.Message);
  }
}
```

Valósítsuk meg a Fahrenheit→Celsius irányú átalakítást. Ennek érdekében készítsünk a jobb oldali nyomógombhoz egy eseménykezelőt, amiben beolvassunk a Fahrenheit szövegmezőben elhelyezett számadatot, átalakítjuk számmá, majd meghívjuk a webszolgáltatás megfelelő metódusát.

```
private void btF2C_Click(object sender, EventArgs e)
```

```
    { try
      { double Fahrenheit = double.Parse(tbFahrenheit.Text);
        double Celsius =
          Konverter.Fahrenheit2Celsius(Fahrenheit);
        tbCelsius.Text = Celsius.ToString();
      }
      catch (FormatException)
      { MessageBox.Show("A megadott adat hibás!");
      }
      catch (System.Net.WebException we)
      { MessageBox.Show(we.Message);
      }
    }
```

Próbáljuk ki az alkalmazást.