Java and C# in depth

Carlo A. Furia, Bertrand Meyer

C#: web service client application example

Workflow

Goal: write a simple C# program that takes an email address from the command line and determines if it is valid

1. Find a (free) web service that offers an email lookup service
2. Get specification for the WS
   - Informal
   - Formal: WSDL
3. Generate a C# stub from the WSDL
4. Compile the stub into a DLL
5. Write the main application, using the service according to its specification
6. Compile the main application and link the DLL to it

A suitable web service

- http://www.cdyne.com/
  offers some (partially) free webservices
- http://wiki.cdyne.com/wiki/index.php?title=Email_Verification
  documents an email verification service
- Download the WSDL with the formal specification from:
  http://ws.cdyne.com/emailverifyws/emailverify.asmx?wsdl

```
<wsdl:definitions targetNamespace="http://ws.cdyne.com/">
   <wsdl:documentation>
    These functions deal with Email Address Verification.  <b>CDYNE advertises a 100% SLA.
    Try to find that kind of SLA from other web service vendors!</b>
   </wsdl:documentation>
   <wsdl:types>
     <s:schema elementFormDefault="qualified" targetNamespace="http://ws.cdyne.com/">
       <s:element name="VerifyMXRecord">
         <s:complexType>
           <s:sequence>
             <s:element minOccurs="0" maxOccurs="1" name="email" type="s:string"/>
...
```

Compile the WSDL into C#

Using the MONO .NET framework v. 4.0

- Input WSDL: **`emailverify.asmx.xml`**
  - **`wsdl emailverify.asmx.xml`**
  - generates: **`EmailVerify.cs`**
- Compile into DLL

  -
    ```
    gmcs EmailVerify.cs
              -target:library
              -r:System,System.Web.Services
    ```

`gmcs`	C# compiler with generics support
`-target:library`	compile to DLL
`-r:libs`	reference to other libraries

- Output: **`EmailVerify.dll`**

Write the main application (1/3)

```
using System;

class EmailVerifier {

  public static void Main(string[] args) {
      // free, but with a limited number of requests
      string testKey = "0";
      if (args.Length == 1) {
            // exactly one argument: the email address
            string addr = args[0];
            // create service client
            EmailVerify s = new EmailVerify();
            // submit request
            int res = s.VerifyMXRecord(addr, testKey);
```

Write the main application (2/3)

```
// interpret the result, according to the spec
switch(res) {
case 0: // invalid address
      Console.WriteLine(addr +
               " is not a valid email address.");
      break;
case -9999: // too many requests!
  Console.WriteLine("Service unreachable.");
  break;
default: // any other value
  Console.WriteLine(addr +
               " is a valid email address.");
  break;
}
```

Write the main application (3/3)

```
    } else {
        // zero or more than one argument
        Console.WriteLine("Invalid syntax.");
    }
  }

}
```

This class is stored in file **`emv.cs`**

Compile the main application

Using the MONO .NET framework v. 4.0

- `gmcs emv.cs -r:EmailVerify.dll`
- generates: `emv.exe`

- Run it
  - `./emv.exe caf@inf.ethz.ch`
  - Output:

```
caf@inf.ethz.ch is a valid email address.
```