ETH Zürich
Chair of Software Engineering

Java and C# in depth

Carlo A. Furia, Bertrand Meyer

Java: web service client application example

Workflow

Goal: write a simple Java 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 Java stub from the WSDL
4. Compile the stub into bytecode
5. Write the main application, using the service according to its specification
6. Compile the main application, referencing the compiled stub

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 Java

Using the JAX-WS framework v. 2.2

- Input WSDL: **`emailverify.asmx.xml`**
  - **`wsimport -keep emailverify.asmx.xml`**
  - **`-keep`** keeps the source **`.java`** files (we will browse them to understand how to use the stub)
- Generates various **`.java`** and **`.class`** files in package **`com.cdyne.ws`**

(Also gives some warnings you can ignore)

Write the main application (1/3)

```
import com.cdyne.ws.*;

public class EMV {

  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
        EmailVerifySoap s = (new
                  EmailVerify ()).getEmailVerifySoap();
        // 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
  System.out.println(addr +
            " is not a valid email address.");
  break;
case -9999: // too many requests!
  System.out.println(“Service unreachable.");
  break;
default: // any other value
  System.out.println(addr +
            " is a valid email address.");
  break;
}
```

Write the main application (3/3)

```
        } else {
            // zero or more than one argument
            System.out.println("Invalid syntax.");
        }
      }

  }
```

This class is stored in file **`EMV.java`**

Compile the main application

Using the JAX-WS framework v. 2.2

- `javac EMV.java`
- generates: `EMV.class`

- Run it
  - `java EMV caf@inf.ethz.ch`
  - Output:

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

Java WS development tools and frameworks

- JAX-WS
- Apache CXF
- Apache Axis2
- Metro
- Spring
- ...