Calling a service from a Java client

How to call a servlet from a Java client and parse XML/Json

How can we call a web api from Java?

We need to connect to the service over HTTP and we may do that using URLConnection:

```
String SERVER_URL_JSON = "http://localhost:8080/userservice?";
String SERVER_URL_XML = "http://localhost:8080/userservicexml?";
String PARAMS = "command=fetch_all";
URL service;
URLConnection con;
String url = null;
if(format.equals("xml")){
  url = SERVER_URL_XML;
}else if(format.equals("json")){
  url = SERVER_URL_JSON;
}else{
  throw new IllegalArgumentException("Format " + format + " is not supported");
}
service = new URL(url+PARAMS);
con=service.openConnection();
```

Then we must read the answer from the service

```
BufferedReader in;
StringBuilder result;
in = new BufferedReader(
       new InputStreamReader(con.getInputStream()));
result = new StringBuilder();
String line;
while( (line = in.readLine())!=null){
  result.append(line);
  result.append("\n");
}
in.close();
```

OK, but now we have either a Json String or XML?

So we need a strategy for formatting Json and XML to Java objects.

We will use the User class from the previous lectures and create a List<User> from the server response (either XML or Json).

One way to do that is to create an interface Formatter which declares two methods common to all formatters. Then we create on XMLFormatter implementation and one JsonFormatter implementation of that interface.

To get the appropriate type of Formatter we use a FormatFactory.

XML response

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<users>
  <user>
    <name>Kalle Anka</name>
    <email>donald@email.dt</email>
    <username>donaldd</username>
  </user>
  <user>
    <name>Joakim von Anka</name>
    <email>scrooge@email.dt</email>
    <username>onkelscrooge</username>
  </user>
  <user>
    <name>Arne Anka</name>
    <email>arne@email.com</email>
    <username>arneanka</username>
  </user>
</users>
```

Json response


```
{
    "users":[
        {
            "Name":"Kalle Anka",
            "Email":"donald@email.dt",
            "UserName":"donaldd"
        },
        {
            "Name":"Joakim von Anka",
            "Email":"scrooge@email.dt",
            "UserName":"onkelscrooge"
        },
        {
            "Name":"Arne Anka",
            "Email":"arne@email.com",
            "UserName":"arneanka"
        }
    ]
}
```

The Formatter interface

```
package client;
import java.util.List;
public interface Formatter{
    public void load(String unformatted);
    public List<User>format();
}
```

The FormatFactory

```
package client;
public class FormatFactory{
    public static Formatter getFormatter(String format){
        if(format!=null && format.equals("xml")){
            return new XMLFormatter();
        }else if(format!=null && format.equals("json")){
            return new JsonFormatter();
    }else return null;
     }
}
```

Why use an interface and a factory?

We want to hide the actual implementation from the client code so that the user decides the format, and the code in the client stays the same regardless of how many formats exist:

```
String format = args[0]; // but check args.length first! - json or xml for instance
GetUsers client = new GetUsers(format);
client.connect();
client.getUsers();
Formatter formatter = FormatFactory.getFormatter(format);
formatter.load(client.result().toString()); ← interface method load
for(User user : formatter.format()){        ← interface method format
  System.out.println(user);
}
```

Parsing XML

To parse some string is to write code that interprets the string so that we can use it in our program.

We'll use the API for XML parsing found in:

javax.xml.parsers

org.w3c.dom

org.w3c.sax

The code for parsing the XML into List‹User›

```
DocumentBuilderFactory factory = DocumentBuilderFactory.newInstance();
DocumentBuilder builder = factory.newDocumentBuilder();
Document document = builder.parse(new ByteArrayInputStream(raw.getBytes()));
List<User>users=new ArrayList<User>();
NodeList nodeList = document.getDocumentElement().getChildNodes();
for (int i = 0; i < nodeList.getLength(); i++) {
  Node node = nodeList.item(i);
  if (node.getNodeType() == Node.ELEMENT_NODE) {
    Element elem = (Element) node;
    String name     = elem.getElementsByTagName("name").item(0).getChildNodes().item(0)
                      .getNodeValue();
    String email    = elem.getElementsByTagName("email").item(0).getChildNodes().item(0)
                       .getNodeValue();
    String userName = elem.getElementsByTagName("username").item(0).getChildNodes().item(0)
                       .getNodeValue();
    User user = User.userBuilder().name(name).email(email).userName(userName).build();
    users.add(user);
  }
}
```

Same thing for Json

```
List<User>users = new ArrayList<User>();
JsonReader reader = Json.createReader( new ByteArrayInputStream(raw.getBytes()) );
JsonStructure jsonStruct = reader.read();
if(jsonStruct.getValueType().equals(OBJECT)){
  JsonObject jo = (JsonObject) jsonStruct;
  JsonArray arr = jo.getJsonArray("users");
  for(JsonValue jv : arr){
    String name     = ((JsonObject)jv).getString("Name");
    String email    = ((JsonObject)jv).getString("Email");
    String userName = ((JsonObject)jv).getString("UserName");
    User user = User.userBuilder()
                    .name(name)
                    .email(email)
                    .userName(userName)
                    .build();
    users.add(user);
  }
}
```

Complete main()

```
public static void main(String[] args){
  try{
    if(args.length!=1){
      System.err.println("USAGE: java client.GetUsers [xml|json]");
    }
    String format = args[0];
    GetUsers client = new GetUsers(format);
    client.connect();  // connect to server
    client.getUsers(); // get the users from the server (as json or xml)
    Formatter formatter = FormatFactory.getFormatter(format);
    formatter.load(client.result().toString());
    System.out.println(client.result());
    for(User user : formatter.format()){
      System.out.println(user);
    }
  }catch(IOException ioe){
    System.err.println("Error getting users: " + ioe.getMessage());
  }
}
```