

HOWTO – KRAK WEB SERVICES I JAVA

1. UDVIKLINGSMILJØ

Eksemplerne heri tager udgangspunkt i to scenarier: Brug af Eclipse¹ og brug af kommandolinien. Ved at se på kommandolinieeksemplerne burde man kunne konfigurere sit yndlings-IDE tilsvarende.

2. AXIS

I disse eksempler bruges Axis til at autogenerere metodestubbe ud fra WSDL-dokumenter. Axis kan downloades fra <http://ws.apache.org/axis/>. I skrivende stund er nyeste stabile version v1.4. Der findes også en v2; den er væsensforskellig anderledes i brug end v1 og kan derfor ikke umiddelbart benyttes sammen med eksemplerne heri.

3. PROJEKTSTRUKTUR

Hvis man da ikke ligefrem selv kompilerer Axis består den downloadede fil af en mængde JAR-filer:

- axis-ant.jar
- axis.jar
- commons-discovery-0.2.jar
- commons-logging-1.0.4.jar
- jaxrpc.jar
- log4j-1.2.8.jar
- saaj.jar
- wsdl4j-1.5.1.jar

Disse kan med fordel lægges i en folder ved navn lib under projektets hovedmappe. Alle senere eksempler i dette dokument vil gå ud fra at de er placeret i en sådan folder. Hvis du bruger Eclipse: Højreklik på projektet i Package Explorer → "Properties" → "Java Build Path" → "Libraries" → "Add External JARs" og tilføj JAR-filerne fra lib-folderen.

4. WSDL → JAVA

For at autogenerere metodestubbe bruger vi det i Axis inkluderede værktøj WSDL2Java. Der er seks forskellige webservices hos Krak, plus den generelle login-service. Dvs. der kan genereres stubbe for følgende dokumenter:

- <http://login.webservice.krak.dk/ticketcentral.asmx?WSDL>
- <http://basicservices.webservice.krak.dk/telesearch.asmx?WSDL>
- <http://basicservices.webservice.krak.dk/companysearch.asmx?WSDL>
- <http://basicservices.webservice.krak.dk/addresssearch.asmx?WSDL>
- <http://basicservices.webservice.krak.dk/routesearch.asmx?WSDL>
- <http://basicservices.webservice.krak.dk/mapping.asmx?WSDL>
- <http://basicservices.webservice.krak.dk/map.asmx?WSDL>

¹ <http://www.eclipse.org/>

Af disse er kun den første obligatorisk for ethvert projekt. Derudover genereres stubbe for de services man har adgang til / brug for. For at generere stubbe med WSDL2Java udføres flg. kommando i en shell fra projektfolderen (én linie, ingen mellemrum mellem semikolonerne):

```
java -cp lib/axis-ant.jar;lib/axis.jar;lib/commons-discovery-0.2.jar;  
lib/commons-logging-1.0.4.jar;lib/jaxrpc.jar;lib/log4j-1.2.8.jar;lib/saaj.jar;  
lib/wsd14j-1.5.1.jar org.apache.axis.wsd1.WSDL2Java <WSDL-fil>
```

Hvis man bruger Linux skal alle semikolon (;) udskiftes med kolon (:) i kommandoen. Der findes også plug-ins til Eclipse der kan køre WSDL2Java, men de vil ikke blive dækket her.

WSDL2Java kan læse både filer på det lokale filsystem og fra URL'er, så man kan bare give den dem af de syv ovenstående URL'er man skal konvertere.

Hvis man bruger Eclipse er det nu blot at trykke F5 med projektet markeret i Package Explorer, og de genererede stubbe dukker op i projektet som pakkerne dk.krak.webservices, Extent.Krak og Extent.Mapping.Krak. Det vil dog umiddelbart være åbenlyst at de genererede stubbe skal masseres lidt for at være brugbare - jeg får her 42 kompileringsfejl og hundredevis af advarsler. Fejlene afhjælpes som følger:

- 1) Find alle tilfælde af Extent.Krak.Point og ret til Point.
- 2) Find alle tilfælde af Extent.Mapping.Krak.Size og ret til Size.
- 3) Find alle tilfælde af \c{Extent.Mapping.Krak.PointInt} og ret til PointInt.

Advarslerne skyldes for en stor del at den autogenererede kode er beregnet til Java 1.4, og jeg bruger her Java 1.5. I Eclipse kan QuickFix ordne de fleste; man kan også blot ignorere dem alt efter temperament og udviklingsstandarder.

4.1 Udfordringer

- Der er pt. en fejl i WSDL-interfacet til Mapping-servicen der får WSDL2Java til at kløjes i xml'en. Derfor: Hvis man skal bruge Mapping, så download WSDL-filen og foretag to ændringer: I linie 500 og 633 skal GetExtent ændres til GetExtent2. Hvis man skal konvertere Mapping-servicen med WSDL2Java angiver man så stien til den lokale, modificerede WSDL-fil.
- Der vil muligvis komme følgende advarsel når WSDL2Java køres: "Unable to find required classes (javax.activation.DataHandler and javax.mail.internet.MimeMultipart). Attachment support is disabled." Dette skyldes at du bruger J2SE i stedet for J2EE, og har ingen praktisk betydning – med mindre du faktisk troede at du brugte J2EE. I så fald peger din PATH på den gale Java-runtime ...
- Mere alvorligt er det at der er et navnesammenfald mellem servicen Map og returtypen Map fra servicen Mapping. Dette betyder at disse to services ikke uden videre kan benyttes i samme projekt, da definitionen af den ene vil overskrive den anden. Problemet overkommes ved først at generere kodelistubben til den ene service med WSDL2Java, omdøbe Map-klassen og så generere næste service. I Eclipse gøres det nemt ved at højreklikke på klassen Map og vælge "Refactor". Giv nu klassen et nyt navn – fx. MapRes – og Eclipse ændrer alle referencer til klassen. Herefter genereres næste WSDL-dokument.

5. ANVENDELSE

Følgende imports tilføjes til de filer der skal kalde Kraks webservices:

```
import java.rmi.RemoteException;  
import javax.xml.rpc.ServiceException;  
import dk.krak.webservice.*;
```

Endvidere hvis man bruger Map-servicen:

```
import Extent.Mapping.Krak.Extent;
import Extent.Mapping.Krak.PointInt;
import Extent.Mapping.Krak.Size;
```

Hvis man vil implementere fejlhåndtering for timeouts etc. kan man også med fordel importere `org.apache.axis.AxisFault`.

5.1 Adresse → Koordinater

```
try
{
    AddressSearch as = new AddressSearchLocator();
    AddressSearchSoap ass = as.getAddressSearchSoap();
    Address[] results =
        ass.parseNSearchAddress("Arne Jacobsens Allé 17, 2300 S");
    if (results.length > 0)
    {
        Address address = results[0];
        Map m = new MapLocator();
        MapSoap ms = m.getMapSoap();
        long rn = address.getRoadNumber();
        int hn = address.getHouseNumberNumericFrom();
        String hc = address.getHouseNumberCharacterFrom();
        AddressResponse response = ms.getCoordinatesByAddress(rn, hn, hc);
        AddressCoordinate coords = response.getAddressCoordinate();
        System.out.println("Atkins bor på UTM-koordinaterne (E" +
            coords.getPlacementLocation().getX() + ", N" +
            coords.getPlacementLocation().getY() + ")");
    }
}
catch (RemoteException e) { e.printStackTrace(); }
catch (ServiceException e) { e.printStackTrace(); }
```

5.2 Kortvisning

Vi fortsætter med de coords vi modtog i foregående afsnit. Observer at enhederne i extentet er i centimeter, mens koordinaterne vi fik var i meter:

```
int centerX = (int)coords.getPlacementLocation().getX() * 100;
int centerY = (int)coords.getPlacementLocation().getY() * 100;
int width = 200000; int height = 200000;
Extent ext = new Extent();
ext.setCenter(new PointInt(centerX, centerY));
ext.setLeft(centerX - width/2);
ext.setRight(centerX + width/2);
ext.setTop(centerY + height/2);
ext.setBottom(centerY - height/2);
ext.setSize(new Size(width, height));
MapResponse mr = ms.getMapByExtent(500, 500, ext);
System.out.println("URL'en til billedet er " + mr.getMapUrl());
```

5.3 Rute på kort

Lad os prøve at finde vej fra Atkins til Krak i Virum. Her går vi skridtet videre og viser billedet i Java:

```

try
{
    AddressSearch as = new AddressSearchLocator();
    AddressSearchSoap ass = as.getAddressSearchSoap();
    Address[] results1 =
        ass.parseSearchAddress("Arne Jacobsens Allé 17, 2300 S");
    Address[] results2 =
        ass.parseSearchAddress("Virumgårdsvej 21, 2830 Virum");
    if (results1.length > 0 && results2.length > 0)
    {
        Address address1 = results1[0];
        Address address2 = results2[0];
        Mapping m = new MappingLocator();
        MappingSoap ms = m.getMappingSoap();
        Map map = ms.getRouteMap
            (
                address1.getRoadNumber(),
                address1.getHouseNumberNumericFrom(),
                address1.getHouseNumberCharacterFrom(),
                address2.getRoadNumber(),
                address2.getHouseNumberNumericFrom(),
                address2.getHouseNumberCharacterFrom(),
                500, 500
            );
        System.out.println("URL'en til rutekortet er " + map.getURL());

        JFrame.setDefaultLookAndFeelDecorated(true);
        JFrame frame = new JFrame("Krak Demo");
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        java.awt.Toolkit tk = Toolkit.getDefaultToolkit();
        Image img = tk.getImage(new URL(map.getURL()));
        JLabel label = new JLabel();
        label.setIcon(new ImageIcon(img));
        frame.getContentPane().add(label);
        frame.setLocation(200, 100);
        frame.pack();
        frame.setVisible(true);
    }
}
catch (RemoteException e) { e.printStackTrace(); }
catch (ServiceException e) { e.printStackTrace(); }
catch (MalformedURLException e) { e.printStackTrace(); }

```

5.4 Kompilering i shell

Vores eksempel ligger som klassen KrakDemo i pakken dk.atkins.gis, så for at kompilere det i shell stiller vi os i projektfolderen og afgiver kommandoen:

```
javac -cp .;lib/axis.jar;lib/jaxrpc.jar dk/atkins/gis/KrakDemo.java
```

Igen – hvis man bruger Linux skal semikolon udskiftes med kolon.