Geografisk lokalisering i JSP

Denne artikel forklarer lidt om hvorfor og hvordan man laver geografisk lokalisering og viser noget kode.

Der er andre artikler for andre web sprog.

Den forudsætter lidt kendskab til generel web, JSP, Java og JDBC.

Skrevet den **15. apr 2009** af **arne_v** I kategorien **Programmering / JSP** | ⭐⭐⭐⭐☆

Historie:
V1.0 - 30/12/2004 - original

Hvad er geografisk lokalisering

Din side finder ud af hvor brugeren er henne. Typisk hvilket land. Men i nogen tilfælde også hvilken landsdel eller hvilken by.

Hvorfor laver man geografisk lokalisering

Det kan der være mange grunde til.

Nogle eksempler:

1) Man vil forbyde adgang til hele siten eller visse sider fra enkelte lande eller eventuelt alle andre lande end ens eget p.g.a. lovmæssige krav.

2) Man vil automatisk skifte til sider i brugerens sprog.

3) Man vil generere sider som viser nærmeste butikker eller nærmeste download server eller lignende.

Hvordan laver man geografisk lokalisering

Man henter adressen på client maskinen fra HTTP headeren Remote-Addr og slår op i en database som mapper fra IP adresser til geografisk lokation.

Kan det ikke gøres nemmere ?

Man kigger på HTTP headeren Accept-Language og ser hvilket sprog browseren er sat til.

I praksis er det ubrugeligt, da masser af ikke personer i ikke engelsk talende lande bruger engelske versioner

af software.

Man laver reverse DNS lookup på adressen på client maskinen og ser hvilket land domæne navnet tilhører.

I praksis er det ubrugeligt, da meget almindelige domæner som .com .org .net .biz etc. ikke nødvendigvis er USA.

Man beder brugeren selv vælge sprog eller nærmeste butik / download site.

Det virker naturligvis for disse formål. Og er absolut en simpel og nem løsning, som man bør overveje.

Løsningen kan åbenlyst ikke bruges til at blokere for uønskede med.

Geo lokaliserings databaser

Man skal som sagt have en database der mapper fra IP adresser til land.

Der eksisterer adskillige sådanne. En masse kommercielle og enkelte gratis.

Det er vigtigt at huske at den database skal løbende opdateres, da internettet ikke er statisk.

Man skal også gøre sig klart at det ikke er en 100% sikker lokalisering.

Forskellene på de kommercielle og de gratis er typisk:
* de gratis er kun på land - de kommercielle er både land og landsdel/by
* de gratis har 90-95% sikkerhed - de kommercielle har 95-98% sikkerhed
* de gratis opdateres en gang om måneden - de kommercielle opdateres hver dag

Jeg kender 2 gratis:
* http://ip-to-country.webhosting.info/
* http://www.maxmind.com/app/geoip_country

Inden du starter brug af dem, så skal du nærlæse deres licens betingelser og checke om de er kompatible med dit brug.

Nedenstående kode viser hvordan de kan bruges. Du vælger hvilken af de 2 du vil bruge og retter eventuelt koden til så den passer til dit formål.

Database byg kode

GeoLoad.java

```
package e_geo;

import java.io.*;
import java.math.*;
import java.net.*;
import java.sql.*;
```

```
import java.util.zip.*;

public class GeoLoad {
    /* START CONFIGURATION */
    private final static String urlWebHostingInfo =
"http://ip-to-country.webhosting.info/downloads/ip-to-country.csv.zip";
    private final static String urlMaxMind =
"http://www.maxmind.com/download/geoip/database/GeoIPCountryCSV.zip";
    private final static String workDir = "C:\\e";
    private final static String dbDriver = "com.mysql.jdbc.Driver";
    private final static String dbUrl = "jdbc:mysql://localhost/Test";
    private final static String dbUsername = "";
    private final static String dbPassword = "";
    /* END CONFIGURATION */
    private static void download(String urlstr, String zipfnm) throws
IOException {
        URL url = new URL(urlstr);
        HttpURLConnection con = (HttpURLConnection) url.openConnection();
        con.connect();
        if (con.getResponseCode() == HttpURLConnection.HTTP_OK) {
           InputStream is = con.getInputStream();
           OutputStream os = new FileOutputStream(workDir + File.separator +
zipfnm);
           byte[] b = new byte[100000];
           int n;
           while ((n = is.read(b)) >= 0) {
              os.write(b, 0, n);
           }
           os.close();
           is.close();
        }
        con.disconnect();
    }
    private static void unzip(String zipfnm, String csvfnm) throws IOException
{
        ZipFile zf = new ZipFile(workDir + File.separator + zipfnm);
        ZipEntry ze = (ZipEntry) zf.entries().nextElement();
        InputStream is = zf.getInputStream(ze);
        OutputStream os = new FileOutputStream(workDir + File.separator +
csvfnm);
        byte[] b = new byte[100000];
        int n;
        while ((n = is.read(b)) >= 0) {
           os.write(b, 0, n);
        }
        os.close();
        is.close();
        zf.close();
    }
    private static void dbimport(String csvfnm, int fromcol, int tocol, int
countrycol, String tblnam) throws ClassNotFoundException, SQLException,
IOException {
        Class.forName(dbDriver);
        Connection con = DriverManager.getConnection(dbUrl, dbUsername,
dbPassword);
```

```
        Statement stmt = con.createStatement();
        try {
            stmt.executeUpdate("DROP TABLE " + tblnam);
        } catch (SQLException e) {
            /* we will end here first time */
        }
        stmt.executeUpdate("CREATE TABLE " + tblnam + "(id INTEGER NOT NULL,
ip1 NUMERIC(10,0), ip2 NUMERIC(10,0), country CHAR(2), PRIMARY KEY(id))");
        stmt.executeUpdate("CREATE INDEX ixip1 ON " + tblnam + "(ip1)");
        stmt.executeUpdate("CREATE INDEX ixip2 ON " + tblnam + "(ip2)");
        stmt.close();
        PreparedStatement pstmt = con.prepareStatement("INSERT INTO " + tblnam
+ " VALUES(?,?,?,?)");
        BufferedReader br = new BufferedReader(new FileReader(workDir +
File.separator + csvfnm));
        int n = 0;
        String line;
        while((line = br.readLine()) != null) {
            n++;
            String[] cols = line.split(",");
            BigDecimal fromval = new BigDecimal(cols[fromcol-1].replace('"', '
').trim());
            BigDecimal toval = new BigDecimal(cols[tocol-1].replace('"', '
').trim());
            String code = cols[countrycol-1].replace('"', ' ').trim();
            pstmt.setInt(1, n);
            pstmt.setBigDecimal(2, fromval);
            pstmt.setBigDecimal(3, toval);
            pstmt.setString(4, code);
            pstmt.executeUpdate();
        }
        pstmt.close();
        con.close();
    }
    private static void loadWebHostingInfo() throws IOException,
ClassNotFoundException, SQLException {
        download(urlWebHostingInfo, "whi.zip");
        unzip("whi.zip", "whi.csv");
        dbimport("whi.csv", 1, 2, 3, "whi");
    }
    private static void loadMaxMind() throws IOException,
ClassNotFoundException, SQLException {
        download(urlMaxMind, "mm.zip");
        unzip("mm.zip", "mm.csv");
        dbimport("mm.csv", 3, 4, 5, "mm");
    }
    public static void main(String[] args) throws IOException,
ClassNotFoundException, SQLException {
        loadWebHostingInfo();
        loadMaxMind();
    }
}
```

Bemærk at dette program gør det hele:
 - downloader zip fil
 - udpakker csv fil fra zip fil
 - loader csv fil til databasen

Du skal kun tilrette konfigurationen i toppen.

Database søge kode

GeoLocate.java

```
package e_geo;

import java.net.*;
import java.sql.*;
import java.util.*;

public class GeoLocate {
    /* START CONFIGURATION */
    private final static String dbDriver = "com.mysql.jdbc.Driver";
    private final static String dbUrl = "jdbc:mysql://localhost/Test";
    private final static String dbUsername = "";
    private final static String dbPassword = "";
    /* END CONFIGURATION */
    private static Connection con;
    private static HashMap cache;
    static {
        try {
            Class.forName(dbDriver);
            con = DriverManager.getConnection(dbUrl, dbUsername, dbPassword);
            cache = new HashMap();
        } catch (ClassNotFoundException e) {
            e.printStackTrace();
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }
    public static String locate(String ip, String tblnam) {
        String country = (String)cache.get(ip);
        if(country == null) {
            try {
                byte[] b = InetAddress.getByName(ip).getAddress();
                long ipnum = 0;
                for(int i = 0; i < 4; i++) {
                    ipnum = (ipnum << 8) | (b[i] & 0xFF);
                }
                synchronized(con) {
                    Statement stmt = con.createStatement();
                    ResultSet rs = stmt.executeQuery("SELECT country FROM " +
tblnam + " WHERE ip1 < " + ipnum + " AND " + ipnum + " < ip2");
                    if(rs.next()) {
                        country = rs.getString(1);
                    } else {
                        country = "??";
```

```
                    }
                    rs.close();
                }
            } catch (SQLException e) {
                e.printStackTrace();
            } catch (UnknownHostException e) {
                e.printStackTrace();
            }
            cache.put(ip, country);
        }
        return country;
    }
}
```

test.jsp

```
<%@ page import="e_geo.*" %>
<%=GeoLocate.locate(request.getRemoteAddr(), "whi")%>
<%=GeoLocate.locate(request.getRemoteAddr(), "mm")%>
```

Du skal selvfølgelig også tilrette konfigurationen her.

GeoLocate.class og JSP siden der bruger den skal deployes helt normalt (ligger udenfor scope af denne artikel).

Den viste test.jsp er naturligvis uinteressant, men den viser hvordan man kalder.

Kommentar af simonvalter d. 31. dec 2004 | 1

Smart med sådan en database... interessant emne der er valgt.

Kommentar af mbm2007 d. 05. jan 2010 | 2