```
package org.spongycastle.p1419a.p1437r.p1438a;

import java.io.IOException;
import java.util.Hashtable;
import java.util.Vector;
import org.apache.commons.p1399io.IOUtils;
import org.spongycastle.p1419a.ASN1Encodable;
import org.spongycastle.p1419a.ASN1ObjectIdentifier;
import org.spongycastle.p1419a.ASN1Primitive;
import org.spongycastle.p1419a.ASN1String;
import org.spongycastle.p1419a.DERUniversalString;
import org.spongycastle.p1419a.p1437r.AttributeTypeAndValue;
import org.spongycastle.p1419a.p1437r.RDN;
import org.spongycastle.p1419a.p1437r.X500NameBuilder;
import org.spongycastle.p1419a.p1437r.X500NameStyle;
import org.spongycastle.util.C30866m;
import org.spongycastle.util.p1479a.Hex;

/* renamed from: org.spongycastle.a.r.a.c */
public class IETFUtils {
    /* renamed from: Y */
    private static int m79405Y(char c) {
        if ('0' <= c && c <= '9') {
            return c - '0';
        }
        return (('a' > c || c > 'f') ? c - 'A' : c - 'a') + 10;
    }

    private static boolean isHexDigit(char c) {
        return ('0' <= c && c <= '9') || ('a' <= c && c <= 'f') || ('A' <= c && c <= 'F')¬
;
    }

    private static String unescape(String str) {
        int i;
        if (str.length() == 0 || (str.indexOf(92) < 0 && str.indexOf(34) < 0)) {
            return str.trim();
        }
        char[] charArray = str.toCharArray();
        StringBuffer stringBuffer = new StringBuffer(str.length());
        if (charArray[0] == '\\' && charArray[1] == '#') {
            i = 2;
            stringBuffer.append("\\#");
        } else {
            i = 0;
        }
        boolean z = false;
        int i2 = 0;
        boolean z2 = false;
        boolean z3 = false;
        char c = 0;
        while (i != charArray.length) {
            char c2 = charArray[i];
            if (c2 != ' ') {
                z3 = true;
            }
            if (c2 == '\"') {
                if (!z) {
                    z2 = !z2;
                } else {
                    stringBuffer.append(c2);
                }
            } else if (c2 == '\\' && !z && !z2) {
                i2 = stringBuffer.length();
                z = true;
                i++;
            } else if (c2 == ' ' && !z && !z3) {
                i++;
            } else if (!z || !isHexDigit(c2)) {
                stringBuffer.append(c2);
            } else {
                if (c != 0) {
                    stringBuffer.append((char) ((m79405Y(c) * 16) + m79405Y(c2)));
                    z = false;
                    c = 0;
                } else {
                    c = c2;
                }
                i++;
            }
            z = false;
            i++;
        }
        if (stringBuffer.length() > 0) {
            while (stringBuffer.charAt(stringBuffer.length() - 1) == ' ' && i2 != stringB¬
uffer.length() - 1) {
                stringBuffer.setLength(stringBuffer.length() - 1);
            }
        }
        return stringBuffer.toString();
    }

    /* renamed from: a */
    public static RDN[] m79411a(String str, X500NameStyle eVar) {
        X500NameTokenizer eVar2 = new X500NameTokenizer(str);
        X500NameBuilder dVar = new X500NameBuilder(eVar);
        while (eVar2.hasMoreTokens()) {
            String nextToken = eVar2.nextToken();
            if (nextToken.indexOf(43) > 0) {
                X500NameTokenizer eVar3 = new X500NameTokenizer(nextToken, '+');
                X500NameTokenizer eVar4 = new X500NameTokenizer(eVar3.nextToken(), '=');
                String nextToken2 = eVar4.nextToken();
                if (eVar4.hasMoreTokens()) {
                    String nextToken3 = eVar4.nextToken();
                    ASN1ObjectIdentifier aqH = eVar.aqH(nextToken2.trim());
                    if (eVar3.hasMoreTokens()) {
                        Vector vector = new Vector();
                        Vector vector2 = new Vector();
                        vector.addElement(aqH);
                        vector2.addElement(unescape(nextToken3));
                        while (eVar3.hasMoreTokens()) {
                            X500NameTokenizer eVar5 = new X500NameTokenizer(eVar3.nextTok¬
en(), '=');

                            String nextToken4 = eVar5.nextToken();
                            if (eVar5.hasMoreTokens()) {
                                String nextToken5 = eVar5.nextToken();
                                vector.addElement(eVar.aqH(nextToken4.trim()));
                                vector2.addElement(unescape(nextToken5));
                            } else {
                                throw new IllegalArgumentException("badly formatted direc¬
tory string");
                            }
                        }
                        dVar.mo186208a(m79413b(vector), m79410a(vector2));
                    } else {
                        dVar.mo186206a(aqH, unescape(nextToken3));
                    }
                } else {
                    throw new IllegalArgumentException("badly formatted directory string"¬
);
```

```
                }
            } else {
                X500NameTokenizer eVar6 = new X500NameTokenizer(nextToken, '=');
                String nextToken6 = eVar6.nextToken();
                if (eVar6.hasMoreTokens()) {
                    dVar.mo186206a(eVar.aqH(nextToken6.trim()), unescape(eVar6.nextToken(¬
)));
                } else {
                    throw new IllegalArgumentException("badly formatted directory string"¬
);
                }
            }
        }
        return dVar.jrR().jrQ();
    }

    /* renamed from: a */
    private static String[] m79410a(Vector vector) {
        int size = vector.size();
        String[] strArr = new String[size];
        for (int i = 0; i != size; i++) {
            strArr[i] = (String) vector.elementAt(i);
        }
        return strArr;
    }

    /* renamed from: b */
    private static ASN1ObjectIdentifier[] m79413b(Vector vector) {
        int size = vector.size();
        ASN1ObjectIdentifier[] oVarArr = new ASN1ObjectIdentifier[size];
        for (int i = 0; i != size; i++) {
            oVarArr[i] = (ASN1ObjectIdentifier) vector.elementAt(i);
        }
        return oVarArr;
    }

    /* renamed from: a */
    public static ASN1ObjectIdentifier m79406a(String str, Hashtable hashtable) {
        if (C30866m.toUpperCase(str).startsWith("OID.")) {
            return new ASN1ObjectIdentifier(str.substring(4));
        }
        if (str.charAt(0) >= '0' && str.charAt(0) <= '9') {
            return new ASN1ObjectIdentifier(str);
        }
        ASN1ObjectIdentifier oVar = (ASN1ObjectIdentifier) hashtable.get(C30866m.toLowerC¬
ase(str));
        if (oVar != null) {
            return oVar;
        }
        throw new IllegalArgumentException("Unknown object id - " + str + " - passed to d¬
istinguished name");
    }

    /* renamed from: bf */
    public static ASN1Encodable m79414bf(String str, int i) throws IOException {
        int length = (str.length() - i) / 2;
        byte[] bArr = new byte[length];
        for (int i2 = 0; i2 != length; i2++) {
            int i3 = (i2 * 2) + i;
            char charAt = str.charAt(i3);
            bArr[i2] = (byte) (m79405Y(str.charAt(i3 + 1)) | (m79405Y(charAt) << 4));
        }
        return ASN1Primitive.m79495aQ(bArr);
    }
    /* renamed from: a */
    public static void m79408a(StringBuffer stringBuffer, RDN bVar, Hashtable hashtable) ¬
{
        if (bVar.jrN()) {
            AttributeTypeAndValue[] jrP = bVar.jrP();
            boolean z = true;
            for (int i = 0; i != jrP.length; i++) {
                if (z) {
                    z = false;
                } else {
                    stringBuffer.append('+');
                }
                m79407a(stringBuffer, jrP[i], hashtable);
            }
        } else if (bVar.jrO() != null) {
            m79407a(stringBuffer, bVar.jrO(), hashtable);
        }
    }

    /* renamed from: a */
    public static void m79407a(StringBuffer stringBuffer, AttributeTypeAndValue aVar, Has¬
htable hashtable) {
        String str = (String) hashtable.get(aVar.jrL());
        if (str != null) {
            stringBuffer.append(str);
        } else {
            stringBuffer.append(aVar.jrL().getId());
        }
        stringBuffer.append('=');
        stringBuffer.append(m79415e(aVar.jrM()));
    }

    /* renamed from: e */
    public static String m79415e(ASN1Encodable fVar) {
        StringBuffer stringBuffer = new StringBuffer();
        int i = 0;
        if (!(fVar instanceof ASN1String) || (fVar instanceof DERUniversalString)) {
            try {
                stringBuffer.append("#" + bytesToString(Hex.encode(fVar.toASN1Primitive()¬
.getEncoded("DER"))));
            } catch (IOException unused) {
                throw new IllegalArgumentException("Other value has no encoded form");
            }
        } else {
            String string = ((ASN1String) fVar).getString();
            if (string.length() <= 0 || string.charAt(0) != '#') {
                stringBuffer.append(string);
            } else {
                stringBuffer.append("\\" + string);
            }
        }
        int length = stringBuffer.length();
        int i2 = 2;
        if (!(stringBuffer.length() >= 2 && stringBuffer.charAt(0) == '\\' && stringBuffe¬
r.charAt(1) == '#')) {
            i2 = 0;
        }
        while (i2 != length) {
            if (stringBuffer.charAt(i2) == ',' || stringBuffer.charAt(i2) == '\"' || stri¬
ngBuffer.charAt(i2) == '\\' || stringBuffer.charAt(i2) == '+' || stringBuffer.charAt(i2) ¬
== '=' || stringBuffer.charAt(i2) == '<' || stringBuffer.charAt(i2) == '>' || stringBuffe¬
r.charAt(i2) == ';') {
                stringBuffer.insert(i2, "\\");
                i2++;
                length++;
            }
            i2++;
        }
        if (stringBuffer.length() > 0) {
            while (stringBuffer.length() > i && stringBuffer.charAt(i) == ' ') {
                stringBuffer.insert(i, "\\");
                i += 2;
            }
        }
        int length2 = stringBuffer.length() - 1;
        while (length2 >= 0 && stringBuffer.charAt(length2) == ' ') {
            stringBuffer.insert(length2, IOUtils.DIR_SEPARATOR_WINDOWS);
            length2--;
        }
        return stringBuffer.toString();
    }

    private static String bytesToString(byte[] bArr) {
        int length = bArr.length;
        char[] cArr = new char[length];
        for (int i = 0; i != length; i++) {
            cArr[i] = (char) (bArr[i] & 255);
        }
        return new String(cArr);
    }

    public static String canonicalize(String str) {
        String lowerCase = C30866m.toLowerCase(str);
        int i = 0;
        if (lowerCase.length() > 0 && lowerCase.charAt(0) == '#') {
            ASN1Primitive decodeObject = decodeObject(lowerCase);
            if (decodeObject instanceof ASN1String) {
                lowerCase = C30866m.toLowerCase(((ASN1String) decodeObject).getString());
            }
        }
        if (lowerCase.length() > 1) {
            while (true) {
                int i2 = i + 1;
                if (i2 < lowerCase.length() && lowerCase.charAt(i) == '\\' && lowerCase.c¬
harAt(i2) == ' ') {
                    i += 2;
                } else {
                    int length = lowerCase.length() - 1;
                }
            }
            int length2 = lowerCase.length() - 1;
            while (true) {
                int i3 = length2 - 1;
                if (i3 > 0 && lowerCase.charAt(i3) == '\\' && lowerCase.charAt(length2) =¬
= ' ') {
                    length2 -= 2;
                } else if (i > 0 || length2 < lowerCase.length() - 1) {
                    lowerCase = lowerCase.substring(i, length2 + 1);
                }
            }
            lowerCase = lowerCase.substring(i, length2 + 1);
        }
        return stripInternalSpaces(lowerCase);
    }

    private static ASN1Primitive decodeObject(String str) {
        try {
            return ASN1Primitive.m79495aQ(Hex.decode(str.substring(1)));
        } catch (IOException e) {
            throw new IllegalStateException("unknown encoding in name: " + e);
        }
    }

    public static String stripInternalSpaces(String str) {
        StringBuffer stringBuffer = new StringBuffer();
        if (str.length() != 0) {
            char charAt = str.charAt(0);
            stringBuffer.append(charAt);
            int i = 1;
            while (i < str.length()) {
                char charAt2 = str.charAt(i);
                if (charAt != ' ' || charAt2 != ' ') {
                    stringBuffer.append(charAt2);
                }
                i++;
                charAt = charAt2;
            }
        }
        return stringBuffer.toString();
    }

    /* renamed from: b */
    public static boolean m79412b(RDN bVar, RDN bVar2) {
        if (bVar.jrN()) {
            if (!bVar2.jrN()) {
                return false;
            }
            AttributeTypeAndValue[] jrP = bVar.jrP();
            AttributeTypeAndValue[] jrP2 = bVar2.jrP();
            if (jrP.length != jrP2.length) {
                return false;
            }
            for (int i = 0; i != jrP.length; i++) {
                if (!m79409a(jrP[i], jrP2[i])) {
                    return false;
                }
            }
            return true;
        } else if (!bVar2.jrN()) {
            return m79409a(bVar.jrO(), bVar2.jrO());
        } else {
            return false;
        }
    }

    /* renamed from: a */
    private static boolean m79409a(AttributeTypeAndValue aVar, AttributeTypeAndValue aVar¬
2) {
        if (aVar == aVar2) {
            return true;
        }
        return aVar != null && aVar2 != null && aVar.jrL().equals(aVar2.jrL()) && canonic¬
alize(m79415e(aVar.jrM())).equals(canonicalize(m79415e(aVar2.jrM())));
    }
}
```