Metacat - Bug #2495

Charset bug: Internationalization

07/19/2006 03:31 PM - Saurabh Garg

Status:	Resolved	**Start date:**	07/19/2006
Priority:	Immediate	**Due date:**	
Assignee:	ben leinfelder	**% Done:**	0%
Category:	metacat	**Estimated time:**	0.00 hour
Target version:	2.0.0	**Spent time:**	0.00 hour
Bugzilla-Id:	2495		

Description

Metacat should be modified in such a way that it can handle characters from other languages also.

Mr. Chau Chin Lin from Taiwan has reported that they have made the following set of changes in Metacat. This enables Metacat to work with 6 languages. The changes are as following:

1.MetacatServlet.java (metacat-src-1.4.0\metacat-1.4.0\src\edu\
ucsb\nceas\metacat\ MetacatServlet.java)

HandleGetOrPost()
if (action.equals("query")) {
/*line:421*/ /*add this line*/response.setContentType("text/xml;
charset=UTF-8");
PrintWriter out = response.getWriter();
handleQuery(out, params, response, username, groupnames,
sess_id);
out.close();

handleReadAction(){
/*line:1030*/ /*add this line*/response.setContentType("text/xml;
charset=UTF-8");
ServletOutputStream out = null;
ZipOutputStream zout = null;
PrintWriter pw = null;
boolean zip = false;

2.build.properties
line 27:jdbc-connect=jdbc:postgresql://localhost/metacat?charSet=UTF-8

3.jsp files(metacat-src-1.4.0\metacat-1.4.0\lib\style\skins\default)
<%@ page contentType="text/html; charset=UTF-8" %>

UTF-8 is enforced as the character encoding for all types of communication.

Also worth noting is the way geoserver does things. It has an entry in web.xml which specifies a filter to encoding conversion

<filter>
<filter-name>Set Character Encoding</filter-name>
<filter-class>org.vfny.geoserver.filters.SetCharacterEncodingFilter</filter-class>
<init-param>
<param-name>encoding</param-name>
<param-value>UTF-8</param-value>
</init-param>
</filter>

A test document with chinese characters can be found here:
http://bugs.tfri.gov.tw/tfri/servlet/metacat?action=read&qformat=default&docid=test100.4.9

A chat log explaing related issues:

[10:05] <sid> the changes which i made for storing all the possible characters in &xxx; form in metacat will probably break things for Lin

[10:06] <sid> i am trying to debug it.. but we will probably need to change a bunch of code later on
[10:10] <matt> yep
[10:12] <sid> this document: http://bugs.tfri.gov.tw/tfri/servlet/metacat?action=read&qformat=xml&docid=test100.4.9
[10:13] <sid> comes back as this document: http://indus.msi.ucsb.edu/knb/metacat?action=read&qformat=xml&docid=sgtest.100.1
[10:14] <matt> if you insert it in 1.6+
[10:14] <matt> ?
[10:14] <sid> yes
[10:14] <matt> with or without their patches?
[10:14] <sid> i havnt tried the patches yet
[10:15] <matt> i think you need them
[10:15] <matt> in order to store the characters in postgres as UTF-8
[10:16] <sid> its mainly because of this code
[10:16] <sid> str.append("&#");
[10:16] <sid> str.append(Integer.toString(ch));
[10:16] <sid> str.append(';');
[10:16] <sid> so any character that we are not familiar with is converted to xx; format
[10:17] <sid> the characters that we are familiar with are the characters in the range of 31 and 128 when converted to int.. newline, carriage return, tab, &, <, >
[10:18] <sid> so that is good enough for most of the documents
[10:19] <sid> but it screws up when we have a character which is not between integer values 0 and 255
[10:19] <sid> which is the case for all other languages
[10:19] <sid> so i can try taking out this code and try setting the encoding to UTF-8 for postgres
[10:21] <sid> so any character that we are not familiar with, we try to store it as it is in metacat
[10:21] <sid> actually in that case i think we can just take away the normalize function
[10:22] <sid> as in maybe we wont need any normalization
[10:23] <sid> but this will probably screw up if the document being inserted has an encoding other than UTF-8
[10:24] <sid> so we will have to enforce that encoding or maybe have an encoding convertor filter

Related issues:

Blocked by FIRST - Bug #3829: Support UTF8 encoded XML in Metacat	**New**	**02/18/2009**

History

#1 - 09/27/2006 08:01 AM - Saurabh Garg

The new relase has broken the code incorporated by the team of Mr. Chau Chin Lin. I have told them that they are welcome to work on the fix as it might be sometime before we get to this. So it might be worth checking with them before someone starts working on this bug. Also I told them if there solution is generic enough, we will be happy to incorporate it into the next release and acknowledge them for it.

#2 - 03/14/2008 02:36 PM - Callie Bowdish

While working with a data set ran into the Chinese measurement called mu (mÇ" or å¸‚ä½©, ç•). According to Wiki it is equal to 666 2/3 mÂ² , ~797.3 sq yd, or
~0.1647 acres. (http://en.wikipedia.org/wiki/Mu_(unit_of_area) Maybe with the Chinese version of Morpho being worked on we may want to become familiar with some common Chinese units of measurement and have a plan for incorporating them into Morpho, EML and Metacat.

#3 - 12/21/2010 02:40 PM - ben leinfelder

I've added encoding detection for incoming/outgoing xml files so that their original encoding is preserved (or defaults to UTF-8).
I've tried to remove most of the printwriters, but there are still many in use when simple XML messages are being returned by the servlet.
We should probably make another pass at these and ensure everything (jsp files included) are defaulting to UTF-8.

For now these actions should be character encoding aware:
read
insert
update
query (UTF-8 to match database)

#4 - 12/23/2010 01:57 PM - ben leinfelder

I've also verified that Replication can handle the encodings (Mac -> Ubuntu tested).

#5 - 12/27/2010 11:21 AM - ben leinfelder

found another spot in query result caching where encodings were being mixed - now using explicit UTF-8

#6 - 01/10/2011 02:57 PM - ben leinfelder

Current state: use XML encoding in prolog if it existing, otherwise default to UTF-8
Next phase: respect encoding specified in HTTP header when receiving XML without an encoding specified in the prolog. We will have to store this information (xml_documents table?) so that it can be used when we return the document with a read request.

#7 - 01/10/2011 02:57 PM - ben leinfelder

Fro reference: http://wiki.apache.org/tomcat/FAQ/CharacterEncoding

#8 - 10/31/2011 05:21 PM - ben leinfelder

Moving this to 2.0.0.

Also, I think relying on the XML-declared encoding is a much safer (easier) bet than inspecting the request header from which it came since the latter is somewhat ephemeral and the former is very clearly saved in the XML file.

#9 - 03/27/2013 02:20 PM - Redmine Admin

Original Bugzilla ID was 2495