WIRIS Scripts – BCLN Documentation

WIRIS script (updated to fix issue with non-unicode characters)

Script from WIRIS = wiris\_safexml\_to\_xml\_backup.php (includes update for non-unicode characters)

Modified script = wr\_modified3.php (has some error feedback)

By default WIRIS formulas are stored as what we call safeXML in Moodle and they are also exported as safeXML. safeXML is a slight modification of MathML replacing some symbols:<http://www.wiris.com/en/plugins/docs/resources/encoding-attributes>

Note: Firefox renders MathML by default, but not Chrome or Edge.

Script can be run from command line or browser:

Browser:

Please note that this script is prepared to be run from the browser.

Within the source code you will see this line:

$file = isset($argv[1])?$argv[1]:(isset($\_GET["file"])?$\_GET["file"]:"");

 As you can see you can load the backup file name using a file GET parameter.

Command Line:

We recommended you to run it from the command line in order to prevent PHP execution timeouts for very large files.

Multiple files:

Instead of iterating through every .mbz file in a directory directly from the wiris\_safexml\_to\_xml script we believe it would be better to build another mini script to iterate through every .mbz file in the directory and call the wiris\_safexml\_to\_xml\_backup.php script once for each file. We believe this is a better approach in order to prevent timeouts (as backups can be huge files and may take a long while to process them).

i.e. You could have a safexml\_to\_xml\_directory.php script that would do the following:

$files = <array\_of\_mbz\_files\_of\_the\_directory>;

foreach ($files as $file) {

     $filename = <filename.mbz>

     //Here it would call the script for example using CURL or file\_get\_contents

     $url  = "wiris\_safexml\_to\_xml\_backup.php?file=".$filename;

     $ch = curl\_init();

     …

     curl\_setopt($ch, CURLOPT\_URL, $url);

     $output = curl\_exec($ch);

     //At this point the output could be saved to a file for logging purposes

     …

}

As supplied from WIRIS (wiris\_safexml\_to\_xml\_backup.php)

<?php

/\*\*

\* Converts safeXML included in a text to XML (MathML)

\* @param string $safexml the text to convert

\* @param string $mode utf8 (default) or html.

\* mode="utf8" converts the characters as they are stored in the DB (use it to convert strings in the database).

\* mode="html" convets the characters using their entities (use it to convert Moodle backups).

\* @param boolean $showstatus show the status (to be used only when converting backups)

\*/

function wiris\_safeXMLtoXML($safexml, $mode="utf8", $showstatus = false) {

$xml = $safexml;

$safexmlsearch = array("«", "»", "§", "¨", "`", "nbsp;");

if ($mode == "html") {

$safexmlreplace = array("&lt;", "&gt;", "&amp;", "\"", "'", "#160;");

}

else {

$safexmlreplace = array("<", ">", "&", "\"", "'", "#160;");

}

if ($safexmlblocks = substr\_count($safexml, "«math ")) {

preg\_match\_all("#(«math.\*?math»)#", $safexml, $matches, PREG\_PATTERN\_ORDER);

if ($showstatus) {

echo " [".$safexmlblocks."]\n";

}

for ($i=0; $i<$safexmlblocks; $i++) {

$tempxml = str\_replace($safexmlsearch, $safexmlreplace, $matches[1][$i]);

$xml = str\_replace($matches[1][$i], $tempxml, $xml);

if ($showstatus) {

echo ".";

}

}

}

return $xml;

}

/\*\*

\* Adjusts some backward compatibility issues

\* @param string $safexml the text

\* @param string $cleansafexml the text clean of backward compatibility issues

\*/

function wiris\_backwardsCompatibilityFilter($safexml) {

$cleansafexml = $safexml;

$search = array ("«mo»§«/mo»");

$replace = array ("«mo»§amp;«/mo»");

$cleansafexml = str\_replace($search, $replace, $safexml);

return $cleansafexml;

}

$file = isset($argv[1])?$argv[1]:(isset($\_GET["file"])?$\_GET["file"]:"");

if (empty($file)) {

echo "\nPlease select a file (i.e. php safeXMLtoXML.php backup.mbz).\n";

}

else {

$fileinfo = pathinfo($file);

if ($fileinfo["extension"] == "mbz") {

//Change the extension to zip in a temp file

$tempzipfile = $fileinfo["filename"]."\_converted.zip";

if (copy($file, $tempzipfile)) {

$zip = new ZipArchive;

$resource = $zip->open($tempzipfile);

$ziptotalfiles = $zip->numFiles;

if ($resource === TRUE) {

for ($i = 0; $i < $ziptotalfiles; $i++) {

$currentfilename = $zip->getNameIndex($i);

if (strstr($currentfilename, "xml")) {

//We only parse XML files in the zip

echo "\n".$currentfilename;

$filecontents = $zip->getFromName($currentfilename);

$filecontents = wiris\_backwardsCompatibilityFilter($filecontents);

$updatedcontent = wiris\_safeXMLtoXML($filecontents, $mode="html", true);

$zip->deleteName($currentfilename);

$zip->addFromString($currentfilename, $updatedcontent);

}

}

$zip->close();

rename($tempzipfile, $fileinfo["filename"]."\_converted.mbz");

}

else {

echo "Error ".$resource." reading ".$tempzipfile." file.";

}

}

}

}

?>