

UNZIPPER

Packt oder Entpackt Dateien.

Status: No .zip or .gz or rar files found. So only zipping functionality available.
Processing Time: 0.00015997886657715 seconds

Archive Unzipper

Select .zip or .rar archive or .gz file you want to extract:

Extraction path (optional):

Enter extraction path without leading or trailing slashes (e.g. "mypath"). If left empty current directory will be used.

Unzip Archive

Archive Zipper

Path that should be zipped (optional):

Enter path to be zipped without leading or trailing slashes (e.g. "zippath"). If left empty current directory will be used.

Zip Archive

Unzipper version: 0.1.0

Contact GitHub API Training Shop Blog About © 2017 GitHub, Inc. Terms Privacy Security Status Help

[unzipper.php](#)

```
<?php
/**
 * The Unzipper extracts .zip or .rar archives and .gz files on
 * web servers.
 * It's handy if you do not have shell access. E.g. if you want to
 * upload a lot
 * of files (php framework or image collection) as an archive to save
 * time.
 * As of version 0.1.0 it also supports creating archives.
 *
 * @author Andreas Tasch, at[tec], attec.at
 * @license GNU GPL v3
 * @package attec.toolbox
 * @version 0.1.0
 */
define('VERSION', '0.1.0');
```

```
$timestart = microtime(TRUE);
$GLOBALS['status'] = array();
$unzipper = new Unzipper;
if (isset($_POST['dounzip'])) {
    //check if an archive was selected for unzipping
    $archive = isset($_POST['zipfile']) ? strip_tags($_POST['zipfile']) :
    '';
    $destination = isset($_POST['extpath']) ?
    strip_tags($_POST['extpath']) : '';
    $unzipper->prepareExtraction($archive, $destination);
}
if (isset($_POST['dozip'])) {
    $zippath = !empty($_POST['zippath']) ? strip_tags($_POST['zippath'])
    : '.';
    // Resulting zipfile e.g. zipper--2016-07-23--11-55.zip
    $zipfile = 'zipper-' . date("Y-m-d--H-i") . '.zip';
    Zipper::zipDir($zippath, $zipfile);
}
$timeend = microtime(TRUE);
$time = $timeend - $timestart;
/**
 * Class Unzipper
 */
class Unzipper {
    public $localdir = '.';
    public $zipfiles = array();
    public function __construct() {
        //read directory and pick .zip and .gz files
        if ($dh = opendir($this->localdir)) {
            while (($file = readdir($dh)) !== FALSE) {
                if (pathinfo($file, PATHINFO_EXTENSION) === 'zip'
                    || pathinfo($file, PATHINFO_EXTENSION) === 'gz'
                    || pathinfo($file, PATHINFO_EXTENSION) === 'rar'
                ) {
                    $this->zipfiles[] = $file;
                }
            }
            closedir($dh);
            if (!empty($this->zipfiles)) {
                $GLOBALS['status'] = array('info' => '.zip or .gz or .rar files
                found, ready for extraction');
            }
            else {
                $GLOBALS['status'] = array('info' => 'No .zip or .gz or rar
                files found. So only zipping functionality available.');
```

```

* Prepare and check zipfile for extraction.
*
* @param $archive
* @param $destination
*/
public function prepareExtraction($archive, $destination) {
    // Determine paths.
    if (empty($destination)) {
        $extpath = $this->localdir;
    }
    else {
        $extpath = $this->localdir . '/' . $destination;
        // todo move this to extraction function
        if (!is_dir($extpath)) {
            mkdir($extpath);
        }
    }
    //allow only local existing archives to extract
    if (in_array($archive, $this->zipfiles)) {
        self::extract($archive, $extpath);
    }
}
/**
* Checks file extension and calls suitable extractor functions.
*
* @param $archive
* @param $destination
*/
public static function extract($archive, $destination) {
    $ext = pathinfo($archive, PATHINFO_EXTENSION);
    switch ($ext) {
        case 'zip':
            self::extractZipArchive($archive, $destination);
            break;
        case 'gz':
            self::extractGzipFile($archive, $destination);
            break;
        case 'rar':
            self::extractRarArchive($archive, $destination);
            break;
    }
}
/**
* Decompress/extract a zip archive using ZipArchive.
*
* @param $archive
* @param $destination
*/
public static function extractZipArchive($archive, $destination) {
    // Check if webserver supports unzipping.
    if (!class_exists('ZipArchive')) {

```

```
$GLOBALS['status'] = array('error' => 'Error: Your PHP version
does not support unzip functionality.');
```

```
    return;
}
$zip = new ZipArchive;
// Check if archive is readable.
if ($zip->open($archive) === TRUE) {
    // Check if destination is writable
    if (is_writable($destination . '/')) {
        $zip->extractTo($destination);
        $zip->close();
        $GLOBALS['status'] = array('success' => 'Files unzipped
successfully');
```

```
    }
    else {
        $GLOBALS['status'] = array('error' => 'Error: Directory not
writeable by webserver.');
```

```
    }
}
else {
    $GLOBALS['status'] = array('error' => 'Error: Cannot read .zip
archive.');
```

```
    }
}
}
/**
 * Decompress a .gz File.
 *
 * @param $archive
 * @param $destination
 */
public static function extractGzipFile($archive, $destination) {
    // Check if zlib is enabled
    if (!function_exists('gzopen')) {
        $GLOBALS['status'] = array('error' => 'Error: Your PHP has no
zlib support enabled.');
```

```
        return;
    }
    $filename = pathinfo($archive, PATHINFO_FILENAME);
    $gzipped = gzopen($archive, "rb");
    $file = fopen($filename, "w");
    while ($string = gzread($gzipped, 4096)) {
        fwrite($file, $string, strlen($string));
    }
    gzclose($gzipped);
    fclose($file);
    // Check if file was extracted.
    if (file_exists($destination . '/' . $filename)) {
        $GLOBALS['status'] = array('success' => 'File unzipped
successfully.');
```

```

    }
    else {
        $GLOBALS['status'] = array('error' => 'Error unzipping file.');
```

```

    }
}

/**
 * Decompress/extract a Rar archive using RarArchive.
 *
 * @param $archive
 * @param $destination
 */
public static function extractRarArchive($archive, $destination) {
    // Check if webserver supports unzipping.
    if (!class_exists('RarArchive')) {
        $GLOBALS['status'] = array('error' => 'Error: Your PHP version
does not support .rar archive functionality. <a class="info"
href="http://php.net/manual/en/rar.installation.php"
target="_blank">How to install RarArchive</a>');
        return;
    }
    // Check if archive is readable.
    if ($rar = RarArchive::open($archive)) {
        // Check if destination is writable
        if (is_writable($destination . '/')) {
            $entries = $rar->getEntries();
            foreach ($entries as $entry) {
                $entry->extract($destination);
            }
            $rar->close();
            $GLOBALS['status'] = array('success' => 'Files extracted
successfully.');
```

```

        }
        else {
            $GLOBALS['status'] = array('error' => 'Error: Directory not
writable by webserver.');
```

```

        }
    }
    else {
        $GLOBALS['status'] = array('error' => 'Error: Cannot read .rar
archive.');
```

```

    }
}

/**
 * Class Zipper
 *
 * Copied and slightly modified from
http://at2.php.net/manual/en/class.ziparchive.php#110719
 * @author umbalaconmeogia
 */
class Zipper {
```

```
/**
 * Add files and sub-directories in a folder to zip file.
 *
 * @param string    $folder
 *   Path to folder that should be zipped.
 *
 * @param ZipArchive $zipFile
 *   Zipfile where files end up.
 *
 * @param int        $exclusiveLength
 *   Number of text to be excluded from the file path.
 */
private static function folderToZip($folder, &$zipFile,
$exclusiveLength) {
    $handle = opendir($folder);
    while (FALSE !== $f = readdir($handle)) {
        // Check for local/parent path or zipping file itself and skip.
        if ($f != '.' && $f != '..' && $f != basename(__FILE__)) {
            $filePath = "$folder/$f";
            // Remove prefix from file path before add to zip.
            $localPath = substr($filePath, $exclusiveLength);
            if (is_file($filePath)) {
                $zipFile->addFile($filePath, $localPath);
            }
            elseif (is_dir($filePath)) {
                // Add sub-directory.
                $zipFile->addEmptyDir($localPath);
                self::folderToZip($filePath, $zipFile, $exclusiveLength);
            }
        }
    }
    closedir($handle);
}
/**
 * Zip a folder (including itself).
 * Usage:
 *   Zipper::zipDir('path/to/sourceDir', 'path/to/out.zip');
 *
 * @param string $sourcePath
 *   Relative path of directory to be zipped.
 *
 * @param string $outZipPath
 *   Relative path of the resulting output zip file.
 */
public static function zipDir($sourcePath, $outZipPath) {
    $pathInfo = pathinfo($sourcePath);
    $parentPath = $pathInfo['dirname'];
    $dirName = $pathInfo['basename'];
    $z = new ZipArchive();
```

```

$z->open($outZipPath, ZipArchive::CREATE);
$z->addEmptyDir($dirName);
if ($sourcePath == $dirName) {
    self::folderToZip($sourcePath, $z, 0);
}
else {
    self::folderToZip($sourcePath, $z, strlen("$parentPath/"));
}
$z->close();
$GLOBALS['status'] = array('success' => 'Successfully created
archive ' . $outZipPath);
}
}
?>

```

```

<!DOCTYPE html>
<html>
<head>
    <title>File Unzipper + Zipper</title>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
    <style type="text/css">
        <!--
        body {
            font-family: Arial, sans-serif;
            line-height: 150%;
        }
        label {
            display: block;
            margin-top: 20px;
        }
        fieldset {
            border: 0;
            background-color: #EEE;
            margin: 10px 0 10px 0;
        }
        .select {
            padding: 5px;
            font-size: 110%;
        }
        .status {
            margin: 0;
            margin-bottom: 20px;
            padding: 10px;
            font-size: 80%;
            background: #EEE;
            border: 1px dotted #DDD;
        }
        .status--ERROR {
            background-color: red;
            color: white;
            font-size: 120%;
        }
    </style>

```

```
}
.status--SUCCESS {
    background-color: green;
    font-weight: bold;
    color: white;
    font-size: 120%
}
.small {
    font-size: 0.7rem;
    font-weight: normal;
}
.version {
    font-size: 80%;
}
.form-field {
    border: 1px solid #AAA;
    padding: 8px;
    width: 280px;
}
.info {
    margin-top: 0;
    font-size: 80%;
    color: #777;
}
.submit {
    background-color: #378de5;
    border: 0;
    color: #ffffff;
    font-size: 15px;
    padding: 10px 24px;
    margin: 20px 0 20px 0;
    text-decoration: none;
}
.submit:hover {
    background-color: #2c6db2;
    cursor: pointer;
}
-->
</style>
</head>
<body>
<p class="status status--<?php echo
strtoupper(key($GLOBALS['status'])); ?>">
    Status: <?php echo reset($GLOBALS['status']); ?><br/>
    <span class="small">Processing Time: <?php echo $time; ?>
seconds</span>
</p>
<form action="" method="POST">
    <fieldset>
```

```
<h1>Archive Unzipper</h1>
<label for="zipfile">Select .zip or .rar archive or .gz file you
want to extract:</label>
<select name="zipfile" size="1" class="select">
  <?php foreach ($unzipper->zipfiles as $zip) {
    echo "<option>$zip</option>";
  }
?>
</select>
<label for="extpath">Extraction path (optional):</label>
<input type="text" name="extpath" class="form-field" />
<p class="info">Enter extraction path without leading or trailing
slashes (e.g. "mypath"). If left empty current directory will be
used.</p>
<input type="submit" name="dounzip" class="submit" value="Unzip
Archive"/>
</fieldset>
<fieldset>
  <h1>Archive Zipper</h1>
  <label for="zippath">Path that should be zipped (optional):</label>
  <input type="text" name="zippath" class="form-field" />
  <p class="info">Enter path to be zipped without leading or trailing
slashes (e.g. "zippath"). If left empty current directory will be
used.</p>
  <input type="submit" name="dozip" class="submit" value="Zip
Archive"/>
</fieldset>
</form>
<p class="version">Unzipper version: <?php echo VERSION; ?></p>
</body>
</html>
Contact GitHub API Training Shop Blog About
© 2017 GitHub, Inc. Terms Privacy Security Status Help
```

From:
<https://wiki.hennweb.de/> - HennWeb

Permanent link:
<https://wiki.hennweb.de/doku.php?id=programmieren:php:programme:unzipper&rev=1601820827>

Last update: 04/10/2020 15:13

