# Prüfungen und Anpassungen am Sourcecode für die Umstellung auf utf-8

## Hinweis

Aufgrund der großen Menge an Funktionen, deren Verhalten sich zusätzlich durch die jeweilige Systemumgebung und teils zahlreichen möglichen Funktionsparameter unterscheidet, können wir die Richtigkeit und Vollständigkeit der hier getätigten Aussagen nicht abschließend garantieren.

## php-String-Funktionen

### **Multibyte Stringfunktionen**

Für die folgenden php-Stringfunktionen gibt es eine Multibytevariante, sie können für die utf-8-Umstellung durch diese Multibytevariante ersetzt werden.

|  |  |
| --- | --- |
| **Funktion / Suchstring** | **Maßnahme** |
| strpos | durch mb\_strpos ersetzen |
| stripos | durch mb\_stripos ersetzen |
| substr | durch mb\_substr ersetzen |
| strlen | durch mb\_strlen ersetzen |
| strstr, strchr | durch mb\_strstr ersetzen |
| stristr | durch mb\_stristr ersetzen |
| strtoupper | durch mb\_strtoupper ersetzen |
| strtolower | durch mb\_strtolower ersetzen |
| chr | durch mb\_chr ersetzen |
| ord | durch mb\_ord ersetzen |
| parse\_str | durch mb\_parse\_str ersetzen |
| mail | durch mb\_send\_mail ersetzen |
| split (deprecated) | durch mb\_split ersetzen |
| str\_split | durch mb\_str\_split ersetzen |
| strrchr | durch mb\_strrchr ersetzen |
| strripos | durch mb\_strripos ersetzen |
| strrpos | durch mb\_strrpos ersetzen |
| substr\_count | durch mb\_substr\_count ersetzen |

**Weitere Stringfunktionen**

Es gibt eine ganze Reihe an PHP-Funktionen (siehe nachfolgende Liste), die keine Multibytevariante haben, sich aber mit unterschiedlichen Zeichensätzen unterschiedlich verhalten können. Diese Funktionen müssen im Einzelfall geprüft werden.

*strval, md5, crc32, crypt, hex2bin, bin2hex, md5\_file, sha1, sha1\_file, urlencode, urldecode, quoted\_printable\_encode, quoted\_printable\_decode, htmlentities, htmlspecialchars, html\_entity\_decode, get\_html\_translation\_table, iconv\_strlen, iconv\_strpos, iconv\_strrpos, iconv\_substr, chunk\_split, convert\_uuencode, convert\_uudecode, count\_chars, similar\_text, printf, sprintf, strcasecmp, strcoll, strncasecmp, strncmp, strrev, strspn, strtr, substr\_compare, fprintf*

Bei den folgenden Funktionen sind vermutlich keine Änderung notwendig:

*strcmp, wordwrap, vsprintf, addcslashes, addslashes, chop, rtrim, explode, implode, join, htmlspecialchars\_decode, lcfirst, ltrim, print, quotemeta, str\_pad, str\_repeat, str\_replace, str\_ireplace, strcspn, strip\_tags, substr\_count, substr\_replace, ucfirst, money\_format, nl\_langinfo, nl2br, number\_format, stripcslashes, stripslashes*

Bisher nicht geprüft wurde die folgende Liste an Funktionen, da sie nicht im Sourcecode des KoGIs-Baukastens vorkommen. Diese sind bei Verwendung ebenfalls zu prüfen und ggf. anzupassen:

*levenshtein, localeconv, hebrev, hebrevc, metaphone, soundex, sscanf, str\_contains, str\_ends\_with, str\_getcsv, str\_rot13, str\_shuffle, str\_starts\_with, str\_word\_count, strnatcasecmp, strnatcmp, strpbrk, strtok, ucwords, vfprintf, vprintf*

## Reguläre Ausdrücke

Bei regulären Ausdrücken muss der Modifier 'u' ergänzt werden (jedes Pattern /.../ → /.../u)

Begründung: Lt. Doku muss /u verwendet werden, sobald String oder Regex-Pattern UTF-8 kodiert sind.

Bei der Verwendung des Modifier ‚u‘ muss sichergestellt sein, dass der String u. das Pattern utf-8-kodiert sind. Bei Bedarf muss hier eine Prüfung des Zeichensatzes und eine bedingte Umwandlung erfolgen.

## Umwandlung Encoding

Bei der Verwendung von PHP-Funktionen, die den Zeichensatz konvertieren, bspw. *utf8\_encode, utf8\_decode, mb\_convert\_encoding, iconv,* iconv\_mime\_decode\_headers, iconv\_mime\_decode, iconv\_mime\_encode, ob\_iconv\_handler, ist sicherzustellen, dass sie auch nach der Umstellung auf utf8 funktionieren. Dies kann bspw. durch geeignete Prüfungen und Bedingungen erreicht werden.