# Kalender mit Bildern der Website

## Inhaltsverzeichnis



Vielleicht taugen die Bilder meiner Website als Kalenderbilder?

Für einen Kalender brauche ich zwölf ausgewählte Bilder und die Kalenderdaten mit den Wochentagen und Feiertagen in den verschiedenen Ländern. Dazu kommt eine Stil-Vorlage, die „online” funktioniert, aber wichtiger ist, dass der Kalender gut ausgedruckt werden kann. Die Kalender müssen dann noch in die Website eingebunden werden.

Die Bildzusammenstellungen kommen aus verschiedenen Quellen. Zu einer Bildergeschichte wählt die Kalender-Ansicht (`style/kal.xslt`) zwölf Bilder aus und verknüpft sie mit der Kalender-Stilvorlage (`kal.xslt`) und passenden Kalenderdaten.

Schöner sind natürlich „von Hand” ausgewählte Kalenderbilder, allein schon deshalb, weil sich die Bildauswahl nicht auf eine Bildergeschichte beschränkt.

Die Kalender gibt es in verschiedenen Sprachen.

Das Bash-Skript `kalender` unterstützt die Präsentation der Kalender auf der Website. Als Gedankenstütze für mich habe ich zusammengestellt, was zu tun ist, wenn wieder ein neues Jahr bevorsteht und Kalender gebraucht werden.

Ich hatte die Kennungen der sprachabhängigen Texte in den Lokalisierungsdateien geändert. An die Platzhalter für die Namen der Monate in den Grunddaten-Dateien hatte ich nicht gedacht. Daher waren in den Kalendern die Namen der Monate verschwunden. Möglicherweise werde ich künftig die Kennungen in den Lokalisierungsdateien ändern. Deshalb beschreibe ich, wie ich den Unfallschaden behoben habe.

# Kalender-Grunddaten

## Grunddaten

Die Kalender-Grunddaten sind die Kalendertage eines Jahres zusammen mit den Tagen des Dezembers des Vorjahres und des Januars des Folgejahres, die Wochentage und Nummern der Woche im Laufe des Jahres sowie die Feiertage.

Wie ist eine Grunddaten-Datei aufgebaut? Die Feiertage sind in der Ländern und Regionen der Welt unterschiedlich. Wie wählt der Server zur Anfrage des Browsers die passenden Grunddaten? Wie kommen die Daten der Feiertage zum Server? Die folgenden Abschnitte versuchen, diese Fragen zu beantworten.

## Aufbau einer Grunddaten-Datei

Die Grunddaten-XML-Dateien nutzen den Namensraum `http://www.w3.org/1999/xhtml`. Sie haben in der Wurzel ein `body`-Element mit 14 `div`-Elementen: das erste `div`-Element mit der ID `d00` für den Dezember des Vorjahres, die folgenden zwölf `div`-Elemente mit den ID `d01` bis `d12` für die Monate des „Hauptjahres", das letzte `div`-Element mit der ID `d13` für den Januar des Folgejahres. Jedes `div`-Element enthält die Bezeichnung des Monats (für 2016 als `h1`-Element) und eine Tabelle mit den Kalendertagen.

Jede Tabellenzeile enthält die Tage von einer oder zwei Kalenderwochen und weitere Zellen für den Monat und die Nummer der Kalenderwoche.

Die Zelle für den Monat steht ganz links. Nur in der ersten Tabellenzeile ist die Bezeichnung des Monats eingetragen, die Zellen darunter bleiben leer. Die Nummer der Kalenderwoche steht in der zweiten Spalte.

Der Anfang der ersten Tabellenzeile ist mit Tagen des Vormonats aufgefüllt, das Ende der letzten Tabellenzeile mit Tagen des Folgemonats. Wörter im Attribut `class` haben die folgende Bedeutung:

`mc`

: Die Zelle enthält die Bezeichnung des Monats oder ist für die Bezeichnung des Monats vorgesehen.

`kw`

: Die Zelle enthält die Nummer der Kalenderwoche.

`p`

: Tag des Vormonats (am Anfang der ersten Tabellenzeile)

`n`

: Tag des Folgemonats (am Ende der letzten Tabellenzeile)

`f`

: Feiertag

`w7`

: Sonntag

Die Wörter `w1`, `w2`, `w3`, `w4`, `w5` und `w6` sind zur Bezeichnunger der übrigen Wochentage vorgesehen, werden aber nicht genutzt.

Wörter im Wert des Attributs `table/@class` bezeichnen den Tabellenaufbau:

`h7`

Eine Tabellenzeile enthält die Tage einer Kalenderwoche.

`h14`

Eine Tabellenzeile enthält die Tage zweier aufeinander folgender Kalenderwochen.

`v7`

Die Tage einer Kalenderwochen stehen in einer Tabellenspalte.

`v7`

Die Tage zweier aufeinander folgender Kalenderwochen stehen in einer Tabellenspalte.

Ich nutze zur Zeit nur den Wert `h7`, Ich kann mir vorstellen, in den Grunddaten für jeden Monat mehrere Tabellen in den verschiedenen Formaten auszugeben und der Websurferin die Wahl zu lassen.

Das Attribut `kb:y` des `body`-Elements enthält das Kalenderjahr. Das Präfix `kb` steht für den Namensraum `http://herbaer.de/xmlns/20151211/kalenderbilder/ [kalenderbilder.rng]`

Die Grunddaten enthalten Platzhalter für die Namen der Monate. Die Platzhalter sind die Elemente `<l:ph id = "`*`ID`*`"/>`. Das Präfix `l` steht für den Namensraum `http://herbaer.de/xmlns/20141210/localization`, der Platzhalter *`ID`* für eine der Monatskennungen `januar`, `februar`, `maerz`, `april`, `mai`, `juni`, `juli`, `august`, `september`, `oktober`, `november`, `dezember`.

# Die Wahl der passenden Grunddaten

Der Browser fordert die Grunddaten unter dem relativen URI `kal/b/`*`JJJJ`*`/a.xml` an. *`JJJJ`* steht für die Jahreszahl. Zur Kalenderansicht der Bildergeschichten ist keine Jahreszahl festgelegt. Der Browser fordert die Grunddaten in diesem Fall unter dem relativen URI `kal/b/a.xml` an. Dieser wird intern umgeleitet zu `kal/b/`*`JJJJ`*`/a.xml`. Dabei steht *`JJJJ`* für das aktuelle Kalenderjahr.

Der Browser teilt in der Regel mit, welche Sprache er bevorzugt (Header `Accept-Language`). Er sendet eine Präferenzliste von Sprachen oder Sprache-/Land-Kombinationen. Ein Eintrag in dieser Liste hat die Form *`LANG-COUNTRY`* oder nur *`LANG`*. *`LANG`* ist die Kennung einer Sprache, *`COUNTRY`* die Kennung eines Landes.

Hier geht es nicht um die Auswahl der Sprache, sondern um die richtigen Feiertage. Eine Grunddaten-Datei ist nicht mit einer bestimmten Sprache verbunden. Zunächst wird der erste Eintrag in der Sprachen-Präferenzliste betrachtet. In manchen Fällen kann man von diesem Eintrag auf eine bestimmte Region innerhalb eines Landes schließen, in der besondere Feiertage gelten, die von den landesweiten Feiertagen abweichen. Einige dieser Fälle sind mir bekannt, und für einige der mir bekannten Fälle habe ich spezielle Grunddaten-Dateien erstellt. Wenn die Datei *`DOCROOT`*`/kal/b/`*`JJJJ/LANG-COUNTRY`*`.xml.` existiert, sendet der Server diese oder die komprimierte Version.

Wenn im ersten Eintrag der Präferenzliste ein Land angegeben ist und die Datei *`DOCROOT`*`/kal/b/`*`JJJJ/COUNTRY`*`.xml.` existiert, sendet der Server diese.

Sonst wird mittels üblicher „Content Negotiation” die zur Sprachenpräferenzliste passende Datei *`DOCROOT`*`/kal/b/`*`JJJJ`*`/a.xml.`*`LANG`*`.` geliefert.

Die Grundlage für die Konfigurationsdatei *`DOCROOT`*`/kal/b/.htaccess` ist die Datei `htaccess`.

# Die Erstellung der Grunddaten

Die Datei `kalender/feiertage/`*`JJJJ/COUNTRY`*`.txt` ist eine einfache Textdatei mit den Feiertagen in der Region, die *`COUNTRY`* bezeichnet. Das Perl-Skript `base.pl` erzeugt daraus nach einfachen Transformationen

die Grunddaten-Datei *DOCROOT*/kal/b/*JJJJ*/*COUNTRY*.xml.. *COUNTRY* kann auch eine Kombination von Sprachcode und Ländercode wie „de-de” sein. Die Ländercodes sind in der Datei laender.dbk aufgeführt.

Die Datei langcodes.dbd nennt viele Sprachen-Codes und zu jeder Sprache ein Land, in dem diese Sprache „zu Hause” ist. Die Transformation langcodes_cmd.xslt liefert aus dieser Datei die Kombinationen von von Sprachcode *LANG* und dem Ländercode *COUNTRY* des Heimatlandes der Sprache. Wenn die Sprache angeboten wird und die Datei *DOCROOT*/kal/b/*JJJJ*/*COUNTRY*.xml. existiert, dann wird diese Datei nach *DOCROOT*/kal/b/*JJJJ*/a.*LANG*.xml. kopiert.

In jedem Verzeichnis *DOCROOT*/kal/b/*JJJJ* wird auch die Datei *DOCROOT*/kal/b/*JJJJ*/a.xml. mit den Kalenderdaten ohne Feiertage angelegt. Diese wird geliefert, wenn keine bessere Wahl gefunden wird.

Zusammen mit jeder Grunddatendatei wird auch die gzip-komprimierte Version der Datei erzeugt.

Das Skript kalender unterstützt die Aufgaben, die mit den Kalendern zusammenhängen.

# kalenderbilder.rng - Bilder für einen Kalender

| | |
|---|---|
| Namespace | http://herbaer.de/xmlns/20151211/kalenderbilder/ |
| Wurzelelement | kalenderbilder |
| (foreign_att) | Attribute anderer XML-Namensräume |
| | *Enthalten in:* kalenderbilder, y, m, t, s, i |
| (anything) | Beliebiger Inhalt |
| | *Enthält:* (anything) (*) |
| | *Enthalten in:* (anything), (foreign_el) |
| (foreign_el) | Elemente anderer XML-Namensräume |
| | *Enthält:* (anything) (*) |
| | *Enthalten in:* kalenderbilder, m, t |
| kalenderbilder | Das Wurzelelement |
| | *Enthält:* (foreign_att), (foreign_el), y, m (+), t |
| | *Enthalten in:* Wurzel |

```
<element name="kalenderbilder">
  <ref name="foreign_att"/>
  <interleave>
    <ref name="foreign_el"/>
    <ref name="el_y"/>
    <oneOrMore>
      <ref name="el_m"/>
    </oneOrMore>
    <ref name="el_t"/>
  </interleave>
</element>
```

| | |
|---|---|
| y | Das Jahr, dem die Bilder zugeordnet sind. |
| | *Enthält:* Datentyp positive_int |
| | *Enthalten in:* kalenderbilder |

```
<element name="y">
  <ref name="foreign_att"/>
  <data type="positive_int"/>
</element>
```

| | |
|---|---|
| m | Daten zum Bild für einen Monat. Ein Bild wird bestimmt durch die Kennung der Bildergeschichte und die Kennung des Bildes innerhalb der Bildergeschichte. |
| | *Enthält:* (foreign_att), @mm, (foreign_el), s, i |
| | *Enthalten in:* kalenderbilder |

```
<element name="m">
  <ref name="foreign_att"/>
  <ref name="att_mm"/>
  <interleave>
    <ref name="foreign_el"/>
    <ref name="el_s"/>
    <ref name="el_i"/>
  </interleave>
</element>
```

| | |
|---|---|
| t | Der Titel des Kalenders. Das Element kann den Titel in mehreren Sprachen enthalten, s. `lt:v [loctext.rng#loctext._el_v]` |
| | *Enthält:* Text, (foreign_att), (foreign_el) (\|) |
| | *Enthalten in:* kalenderbilder |

```
<element name="t">
  <ref name="foreign_att"/>
  <choice>
    <ref name="foreign_el"/>
    <text/>
  </choice>
</element>
```

@mm — Die Nummer des Kalendermonat in zwei Dezimalziffern. `01` für Januar, `12` für Dezember.

*Enthalten in:* m

s — Die Kennung der Bildergeschichte, der das Bild entnommen ist.

*Enthält:* Datentyp word

*Enthalten in:* m

```
<element name="s">
  <ref name="foreign_att"/>
  <data type="word"/>
</element>
```

i — Die Kennung des Bildes.

*Enthält:* Datentyp word

*Enthalten in:* m

```
<element name="i">
  <ref name="foreign_att"/>
  <data type="word"/>
</element>
```

# Sonderfälle der Sprache-/Land-Kombination

Manche Kombinationen von Sprachcode und Ländercode lassen auf eine Region innerhalb des Landes schließen, in der besondere Feiertage gelten, die von den landesweiten Feiertagen abweichen. Die folgende Tabelle enthält die Kennungen, für die es eigene Kalendergrunddaten-Dateien gibt.

Die Zuordnung einer Sprache zu einem Landesteil ist mitunter schwierig. Katalanisch oder eine Variante des Katalanischen wird in mehreren spanischen autonomen Gemeinschaften gesprochen. Ich ordne hier der katalanischen Sprache Katalonien zu. Aranesisch wird vor allem in Katalonien gesprochen, daher ordne ich Aranesisch ebenfalls Katalonien zu.

Einige Kombinationen von Sprachcode und Ländercode lassen auf einen Landesteil schließen, in dem es aber keine besonderen Feiertage gibt. Dieses Kombinationen sind hier nicht aufgeführt.

| | | |
|---|---|---|
| `cy-UK` | Walisisch / Vereinigtes Königreich | Feiertage in Wales |
| `gd-UK` | Schottisch-Gälisch / Vereinigtes Königreich | Feiertage in Schottland |
| `ga-UK` | Irisch / Vereinigtes Königreich | Feiertage in Nordirland |
| `ca-ES` | Katalanisch / Spanien | Feiertage der Autonomen Gemeinschaft Katalonien |
| `eu-ES` | Baskisch / Spanien | Feiertage der Autonomen Gemeinschaft Baskenland |
| `gl-ES` | Galicisch / Spanien | Feiertage der Autonomen Gemeinschaft Galicien |
| `fr-CA` | Französisch / Kanada | Feiertage in der Provinz Quebec |
| `gd-CA` | Schottisch-Gälisch / Kanada | Feiertage in der Provinz Neuschottland |
| `fy-DE` | Friesisch / Deutschland | Feiertage in Niedersachsen |
| `da-DE` | Dänisch / Deutschland | Feiertage in Schleswig-Holstein |

# Datei htaccess

```
# file KLEIDER/web/src/kalender/htaccess
# <?install location = "kal/b/.htaccess"?>
# 2020-12-18 Länderkennung in Klein- oder Großbuchstaben


# Diese Direktive soll sicherstellen,
# dass die Datei JAHR.xml geliefert wird,
# wenn es keine passende Sprachversion gibt.
ForceLanguagePriority None


RewriteEngine On
RewriteBase /kal/b/


# RewriteMap ist im .htaccess-Kontext nicht möglich.


# a. wird zum aktuellen Jahr umgeleitet
RewriteRule ^a\.(.*)$  %{TIME_YEAR}/a.$1 [L]


# Ist eine Länderkennung bereits geprüft?
RewriteRule !/a\. -  [L]


RewriteCond %{HTTP:Accept-Language} ^([a-z]+-?([A-Za-z]*)) [NC]
RewriteRule ^ - [E=XLANGCNT:%1,E=XCNT:%2]


RewriteCond %{DOCUMENT_ROOT}/kal/b/$1/%{ENV:XLANGCNT}.xml. -s
RewriteRule ^([0-9]*)/a $1/%{ENV:XLANGCNT}.xml [L]


RewriteCond %{DOCUMENT_ROOT}/kal/b/$1/%{ENV:XCNT}.xml. -s
RewriteRule ^([0-9]*)/a $1/%{ENV:XCNT}.xml [L]
```

# base.pl

[Quelltext]

# Übersicht

`base.pl --help|--version`

`base.pl` [ `--verbose` … | `--no_verbose` ]
[ `--country` *`COUNTRY`* ]… [ `--out` *`OUT`* ] [ `--year` *`YEAR`* ]
[ `--f` *`F`* ]… [ `--w` *`W`* ]… [ `--ft` *`FT`* ]
[ `--mode` { `h7` | `h14` | `v7` | `v14` } ]…
[ `--monmode` { `de` | `num` | `i` } ]
[ `--wd` | `--no_wd` ] [ `--moncol` | `--no_moncol` ] [ `--weekno` | `--no_weekno` ]

# Optionen

`--help`

Gibt eine kurze Hilfe aus

`--version`

Gibt kurze Hinweise zum Programm und die Version aus.

`--verbose`

Erhöht den Umfang der Meldungen nach STDERR.

`--no_verbose`

Unterdrückt die Ausgabe von Meldungen. Die Optionen `--verbose` und `--no_verbose` werden der Reihe nach ausgewertet.

`--country` *`COUNTRY`*

Der Wert *`COUNTRY`* ersetzt den Platzhalter *`${ctry}`* im Pfad der Ausgabedatei *`OUT`*. Wenn das Argument mehrfach verwendet wird, wird für jeden Wert *`COUNTRY`* eine Ausgabedatei erstellt. *`COUNTRY`* ist hier die Kennung eines Landes oder einer Region.

`--out` *`OUT`*

*`OUT`* ist der Pfad einer Ausgabedatei. Er kann die Platzhalter *`${year}`* für das Kalenderjahr und *`${ctry}`* für die Kennung des Landes (*`COUNTRY`*) enthalten sowie die Platzhalter *`${monmode}`*, *`${[cnt]wd}`*, *`${[cnt]moncol}`*, *`${[cnt]weekno}`* für die entsprechenden Befehlszeilenargumente Bei mehreren Ländern sollte er den Platzhalter *`${cntry}`* enthalten.

`--year` *`YEAR`*

Das Kalenderjahr. Ein Kalenderjahr, das aus der Eingabedatei *`FT`* gelesen wird, hat Vorrang vor *`YEAR`*.

`--f` *`F`*

*`F`* ist das Datum einer Feiertags im Format `MM-TT`. Das Argument kann für mehrere Feiertage wiederholt werden. Das Argument wird ignoriert, wenn Feiertage aus der Texteingabe *`FT`* gelesen werden. Wenn dieses Argument nicht benutzt wird und auch keine Feiertage aus *`FT`* gelesen werden, dann werden die gesetzlichen Feiertage in

der Bundesrepublik Deutschland als Feiertage markiert (die „Standard”-Feiertage). Das Argument `-f x` ohne Eingabe weiterer Feiertage bewirkt, dass in der Ausgabe keine Feiertage markiert werden.

`--w` *W*

*W* ist das Datum eines Feiertags im Format `MM-TT`. Das Argument kann für mehrere Feiertage wiederholt werden. Zusätzlich gelten die „Standard”-Feiertage es sei denn, auch die Option `--f` wird benutzt (nicht ratsam).

Wenn Feiertage aus der Texteingabe *FT* gelesen werden, werden die Feiertage in der Befehlszeile (*F* oder *W*) ignoriert.

`--ft` *FT*

Die Feiertage und das Jahr können aus einer Texteingabe gelesen werden. Die Optionen `--f` und `--w` haben dann keine Wirkung.

Wenn *FT* `STDIN` ist, wird die Standard-Eingabe gelesen.

Wenn *FT* `none` oder nicht definiert ist, bestimmen die Argumente *F* und *W* die Feiertage.

Sonst ist *FT* der Dateipfad einer Textdatei, aus der die Feiertage und möglicherweise das Jahr gelesen werden.

Die Datei wird zeilenweise gelesen. Leerzeichen am Zeilenanfang und Zeileneende werden ignoriert. Zeilen, die mit dem Zeichen # beginnen, sind Kommentarzeilen. Zeilen, die das Format `MM-TT` oder `MM-TT # Kommentar` haben, bezeichnen einen Feiertag. Eine Zeile im Format `20[0-9][0-9]` gibt ein Jahr an. Dieser Wert hat Vorrang vor dem Befehlszeilen-Argument *YEAR*. Die Eingabe endet, wenn eine Zeile mit `#END Feiertage` beginnt.

`--mode` *MODE*

*MODE* bestimmt die Anordnung der Kalendertage auf einem Monatsblatt. Die möglichen Werte sind:

| | |
|---|---|
| `h14` | Die Tage zweier Wochen werden in einer waagerechten Reihe ausgegeben. |
| `h7` | Die Tage einer Woche werden in einer waagerechten Reihe ausgegeben. |
| `v14` | Die Tage zweier Wochen werden in einer senkrechten Spalte ausgegeben. |
| `v7` | Die Tage einer Woche werden in einer senkrechten Spalte ausgegeben. |

Das Argument kann für mehrere Tabellen je Monat wiederholt werden. Die Tabellen sind durch das Attribut `class` gekennzeichnet.

`--monmode` *MONMODE*

*MONMODE* bestimmt der Art der Angabe der Monate. Die möglichen Werte sind:

| | |
|---|---|
| `de` | Deutsche Monatsnamen |
| `num` | Format `MM-JJJJ`. |
| `i` | Platzhalter `<l:ph id = "`*ID*`"/>`, s. Grunddaten |

`--wd`

Alle Wochentage werden durch ein Wort im Atttribut `td/@class` gekennzeichnet. Sonst werden nur Sonntage gekennzeichnet.

`--no_wd`

Hebt die Wirkung der Option `--wd` auf.

`--moncol`

Es wird eine Spalte oder Zeile für die Bezeichnung des Monats ausgegeben.

`--no_moncol`

Hebt die Wirkung der Option `--moncol` auf.

`--weekno`

Es wird eine Spalte oder Zeile mit der Wochennummer ausgegeben.

`--no_weekno`

Hebt die Wirkung der Option `--weekno` auf.

# Beschreibung

Dieses Programm gibt die Kalender-Grunddaten für ein Jahr aus.

# Einschränkungen

In einigen Ländern beginnt die Woche nicht mit dem Montag. In manchen Ländern werden die Wochen des Jahres anders gezählt.

# Software-Voraussetzungen

Das Programm ist mit Perl Version 5.10.1 entwickelt. Es benutzt die folgenden Module:

`Herbaer::Readargs`

Die Funktion `read_args` aus diesem Modul verarbeitet die Befehlszeilenargumente, die Funktion `print_message_with_values` gibt die Hilfe mit den aktuellen Einstellungen aus.

`Herbaer::Replace`

Die Funktion `replace` aus diesem Modul ersetzt die Platzhalter im Wert *`OUT`*.

# Quellen

Die Formel zur Berechnung des Kalenderdatums des Osterfestes habe ich der Website `http://www.gmarts.org/index.php?go=415` entnommen. Danach richten sich die Bestimmung der Wochentage und der Daten der gesetzlichen Feiertage des Osterfestkreises.

Für Hinweise zu Formeln zur Berechnung der Daten anderer besonderer Tage (zum Beispiel des Jahresbeginns nach dem chinesischen Mondkalender) bin ich dankbar.

Hinsichtlich der Wochenzählung und des ersten Tags der Woche orientiere ich mich am Wikipedia-Artikel `https://de.wikipedia.org/wiki/ISO_8601`

# Quelltext

[Beschreibung]

```
#!/usr/bin/perl -w
# Basis-Kalenderdaten
# 2015-12-16 Herbert Schiemann <h.schiemann@herbaer.de>


# 2016-11-01 Voreinstellung --year
# 2016-11-23 Voreinstellung --monmode i
# 2016-12-07 --lang: mehrere Sprachen mit mehreren Ausgabedateien
# 2017-05-25 --country statt --lang, Voreinstellung --out
# 2017-05-29 --ft: Feiertage aus Textdatei
# 2017-05-31 Jahr aus FT lesen
# 2020-04-12 Voreinstellung --out


use utf8;               # Dieser Quelltext ist utf-8-kodiert
use Cwd qw(realpath);
use File::Spec::Functions;
use Herbaer::Readargs;   # read_args ()
use Herbaer::Replace;    # replace ()
use POSIX;


binmode (STDERR, ":encoding(utf-8)");


my $args = {
   "[cnt]verbose" => 1,
   "country"      => [],     # Ländercodes
   "out"          => undef,  # Ausgabedatei mit Platzhaltern
   "year"         => undef,
   "f"            => [],     # vorgegebene Feiertage
   "w"            => [],     # weitere Feiertage
   "ft"           => undef,  # Feiertage aus Textdatei
   "mode"         => [],     # Liste der Tabellenformate
   "monmode"      => "i",    # de:  deutsche Monatsnamen
                             # num: Zahlen
                             # i:   Platzhalter
   "[cnt]wd"      => undef,  # alle Wochentage ausgeben
   "[cnt]moncol" => 1,       # Spalte für Monatsnamen
   "[cnt]weekno" => 1,       # Wochenzahlen ausgeben
};


my $data = {
   "monate" => [
      # 0  Monatsname,
      # 1  Zahl der Tage des Monats
      # 2  Lfd. Nummer des 1. Tages ab 1. Januar (1),
      # 3  Wochentag des Monatsersten von 1 (Mo) bis 7 (So)
      ["Dezember",  31, 1, 1],
      ["Januar",    31, 1, 1],
      ["Februar",   28, 1, 1],
      ["März",      31, 1, 1],
      ["April",     30, 1, 1],
      ["Mai",       31, 1, 1],
      ["Juni",      30, 1, 1],
      ["Juli",      31, 1, 1],
      ["August",    31, 1, 1],
      ["September", 30, 1, 1],
      ["Oktober",   31, 1, 1],
      ["November",  30, 1, 1],
      ["Dezember",  31, 1, 1],
      ["Januar",    31, 1, 1],
      ],
   # Wochentag des 1. Januar, 1 Montag, 7 Sonntag
   "wotagstart" => undef,
   # Anzahl der Kalenderwochen
   "aw"         => undef,
   # Anzahl der Kalenderwochen des Vorjahres
   "awv"        => undef,
   # Hash der Feiertage: Schlüssel: Datums-Zeichenkette dd-mm
   "f"          => undef,
};
```

```
sub set_default {
   my $args = shift;
   my $verb = $args -> {"[cnt]verbose"};
   if (! $args -> {"out"}) {
      my $b = realpath ($0);
      $b =~ s/\/src\/kalender\/base\.pl//;
      $args -> {"out"} = "$b/kalender/base/\${year\}/\${ctry\}.xml",
   }
   # Feiertage
   my $ft = $args -> {"ft"};
   my $file_read = 0;
   if ($ft && $ft ne "none") {
      my $f = [];
      my $l;
      my $h;
      if ( $ft eq "STDIN" ) {
         $h = STDIN
      }
      elsif ($ft) {
         open ($h, "<:encoding(utf-8)", $ft) or do {
            print "Kann Datei \"$ft\" nicht lesen: $!\n" if $verb;
         };
      }
      if ($h) {
         while (defined ($l = <$h>)) {
            last if $l =~ /^\s*#END\s+Feiertage/;
            $l =~ s/#.*$//;
            $l =~ s/\s+$//;
            $l =~ s/^\s+//;
            next unless $l;
            if ( $l =~ /^\s*([01][0-9]-[0-3][0-9])$/ ) {
               push (@$f, $1);
            }
            elsif ( $l =~ /(20[0-9]{2})$/ ) {
               $args -> {"year"} = 0 + $1;
            }
         }
         $args -> {"f"} = $f;
         $args -> {"w"} = [];
         $file_read = 1;
         close $h unless $ft eq "STDIN";
      }
   }
   if (! $args -> {"year"}) {
      my $m;
      my $y;
      my $t = [localtime()];
      $m = $t -> [4];
      $y = $t -> [5] + 1900;
      $y += 1 if $m > 9;
      $args -> {"year"} = $y;
   }
   if (! @{$args -> {"country"}} ) {
      push (@{$args -> {"country"}}, "de");
   }
   init_monate ($args, $data);
   if (!$file_read && ! @{$args -> {"f"}}) {
      $args -> {"f"} = standard_feiertage ($args);
   }
   if (! @{$args -> {"mode"}}) {
      $args -> {"mode"} = ["h7"];
   }
   $args -> {"[cnt]wd"} //= 0;
} # set_default


# Wochentag des 1. Januar
sub wotagstart {
   my $year   = shift;
   my $ostern = datestring_ostern ($year);
   $ostern =~ /^(\d\d)-(\d\d)$/;
   my $mth = $1 + 0;
   my $dy  = $2 + 0;
   $dy += 31 if $mth > 1;
   if ($mth > 2) {
      $dy += 28 + (
         $year % 400 == 0 ? 1 :
         $year % 100 == 0 ? 0 :
         $year %   4 == 0 ? 1 :
         0
      );
   }
   $dy += 31 if $mth > 3;
   $dy += 30 if $mth > 4;
   $dy += 31 if $mth > 5;
   (700 - $dy) % 7 + 1;
} # wotagstart
```

```
# Anzahl der Kalenderwochen eines Jahres
sub anzwochen {
   my $year = shift;
   my $dy = 365 + (
      $year % 400 == 0 ? 1 :
      $year % 100 == 0 ? 0 :
      $year %   4 == 0 ? 1 :
      0
   );
   floor ( ($dy + (wotagstart ($year) + 2) % 7 ) / 7);
} # anzwochen


sub init_monate {
   my ($args, $data) = @_;
   my $monate = $data -> {"monate"};
   my $m;      # Monat als Zahl von 0 (Dezember Vorjahr) bis 13 (Januar Folgejahr)
   $m = 0;
   $monate -> [$m] -> [2] = 1 - 31;
   my $year = $args -> {"year"};
   # Tage des Februar korrigieren
   $m = 2;
   $monate -> [$m] -> [1] +=
      $year % 400 == 0 ? 1 :
      $year % 100 == 0 ? 0 :
      $year %   4 == 0 ? 1 :
      0;
   for ($m = 1; $m < 14; ++$m) {
      # 2  Lfd. Nummer des 1. Tages ab 1. Januar,
      $monate -> [$m] -> [2] = $monate -> [$m - 1] -> [2] + $monate -> [$m - 1] -> [1];
   }
} # init_monate


sub init_data {
   my ($args, $data) = @_;
   my $verbose = $args -> {"[cnt]verbose"};

   my $year = $args -> {"year"};
   $data -> {"wotagstart"} = wotagstart ($year);
   $data -> {"aw"}         = anzwochen  ($year);
   $data -> {"awv"}        = anzwochen  ($year - 1);

   # Wochentag des Monatsersten
   my $monate = $data -> {"monate"};
   my $mon;
   for $mon (@$monate) {
      # 2  Lfd. Nummer des 1. Tages ab 1. Januar,
      # 3  Wochentag des Monatsersten von 1 (Mo) bis 7 (So)
      $mon -> [3] = ($mon -> [2] + $data -> {"wotagstart"} + 33) % 7 + 1;
   }

   # Feiertage
   my $f = {};
   $data -> {"f"} = $f;
   for $mon (@{$args -> {"f"}}, @{$args -> {"w"}}) {
      $f -> {$mon} = 1;
   }
} # init_data


sub lnr_to_datestring {
   my $lnr = shift;
   my $monate = $data -> {"monate"};
   my $m = 1;
   while ( $monate -> [$m] -> [1] + $monate -> [$m] -> [2] - 1 < $lnr && $m < 13) {
      ++$m;
   }
   sprintf ("%02d-%02d", $m, $lnr - $monate -> [$m] -> [2] + 1);
} # lnr_to_datestring


sub datestring_ostern {
   my $year = shift;

   # Ostern 2016: 5. April
   # von http://www.gmarts.org/index.php?go=415
   my $a = floor ($year / 100);
   my $b = $year % 100;
   my $c = floor (3 * ($a + 25)) / 4;
   my $d = (3 * ($a + 25)) % 4;
   my $e = floor ((8 * ($a + 11)) / 25);
   my $f = (5 * $a + $b) % 19;
   my $g = (19 * $f + $c - $e) % 30;
   my $h = floor (($f + 11 * $g) / 319);
   my $j = floor ((60 * (5 - $d) + $b) / 4);
   my $k = (60 * (5 - $d) + $b) % 4;
   my $m = (2 * $j - $k - $g + $h) % 7;
   # Monat und Tag des Osterfestes
   my $mth = floor (($g - $h + $m + 114) / 31);
   my $dy  = ($g - $h + $m + 114) % 31 + 1;
   sprintf ("%02d-%02d", $mth, $dy);
} # datesting_ostern

=for comment
```

```
Function EasterHodges(dy, mth, ByVal y, ByVal method) As Boolean


'by David Hodges, derived by refining the "Butcher's Ecclesiastical Calendar" rule
'eliminating one step in the process


Dim a, b, c, d, e, f, g, h, j, k, m, n, p
' Validate arguments
    If method <> 3 Or y < 1583 Or y > 4099 Then
        EasterHodges = False
        d = 0
        m = 0
        MsgBox "Hodges method only applies to the revised calculation in the Gregorian calendar from 1583 to 4099 AD"
        Exit Function
    End If
    EasterHodges = True
    a = y \ 100
    b = y Mod 100
    c = (3 * (a + 25)) \ 4
    d = (3 * (a + 25)) Mod 4
    e = (8 * (a + 11)) \ 25
    f = (5 * a + b) Mod 19
    g = (19 * f + c - e) Mod 30
    h = (f + 11 * g) \ 319
    j = (60 * (5 - d) + b) \ 4
    k = (60 * (5 - d) + b) Mod 4
    m = (2 * j - k - g + h) Mod 7
    n = (g - h + m + 114) \ 31
    p = (g - h + m + 114) Mod 31
    dy = p + 1
    mth = n
'Easter Sunday is g - h + m days after March 22nd
'(the earliest possible Easter date)
End Function


=cut


# Standard-Feiertage
sub standard_feiertage {
   my $args   = shift;
   my $year   = $args -> {"year"};
   my $monate = $data -> {"monate"};
   my $list = [
    "01-01", # Neujahr
    "13-01", # Neujahr Folge-Januar
    "05-01", # Maifeiertag
    "10-03", # Tag der deutschen Einheit
    "11-01", # Allerheiligen
    "12-25", # Weihnachten
    "12-26",
    "00-25", # Weihnachten Vor-Dezember
    "00-26",
   ];

   my $ostern = datestring_ostern ($year);
   $ostern =~ /^(\d\d)-(\d\d)$/;
   my $mth = $1 + 0;
   my $dy  = $2 + 0;
   my $ostern_lnr = $dy + $monate -> [$mth] -> [2] - 1;
   push (@$list,
         lnr_to_datestring ($ostern_lnr - 2),  # Karfreitag
         lnr_to_datestring ($ostern_lnr),      # Ostern
         lnr_to_datestring ($ostern_lnr + 1),  # Ostermontag
         lnr_to_datestring ($ostern_lnr + 39), # Himmelfahrt
         lnr_to_datestring ($ostern_lnr + 49), # Pfingsten
         lnr_to_datestring ($ostern_lnr + 50),
      );
   $list;
} # standard_feiertage


# gibt die Version nach STDOUT aus
sub version {
   print << 'VERSION';
KLEIDER/web/src/kalender/base.pl
Basis-Kalenderdaten
2020-04-12
2015 - 2020 Herbert Schiemann <h.schiemann@herbaer.de>
VERSION
};
$args -> {"[sr]version"} = sub { version (); exit 0; };
```

```
$args -> {"[sr]help"} = sub {
   set_default ($args);
   version ();
   print_message_with_values (<<"HELP", $args);
$0 --help     zeigt diese Hilfe an
$0 --version  zeigt die Programm-Version an

$0 [option]...
--[no_]verbose           Meldungen nach STDERR ausgeben \${[cnt]verbose}
--country    COUNTRY     Kennungen des Landes \${country}
--out        OUT         Ausgabedatei mit Platzhaltern
                         \${out}
--year       YEAR        Kalenderjahr \${year}
--f          F ..        Standard-Feiertage
                         \${f}
--w          W ..        weitere Feiertage
                         \${w}
--ft         FT          Feiertage aus Textdatei FT
                         \${ft}
--mode       MODE ..     Tabellenformate \${mode}
--monmode    MONMODE     de, num oder i \${monmode}
--[no_]wd                Wochentage ausgeben \${[cnt]wd}
--[no_]moncol            Spalte der Monatsnamen \${[cnt]moncol}
--[no_]weekno            Spalte der Wochennummern \${[cnt]weekno}
HELP
   exit 0;
}; # help


read_args   ($args);
set_default ($args);
init_data      ($args, $data);
print_kalender ($args, $data);


sub print_kalender {
   my ($args, $data) = @_;
   my $year = $args -> {"year"};
   my $obase = $args -> {"out"};
   my $fh;
   my $out;
   my $country;
   my $m;
   for $country (@{$args -> {"country"}}) {
      $args -> {"ctry"} = $country;
      $fh  = undef;
      $out = replace ($obase, $args);
      print "Ausgabedatei $out\n" if $args -> {"[cnt]verbose"};
      open ($fh, ">:encoding(utf-8)", $out) or do
      {
         print STDERR "Kann Datei \"", $out, "\" nicht öffnen: ", $!, "\n"
            if $args -> {"[cnt]verbose"};
         exit;
      };
      $data -> {"fh"} = $fh;
      print $fh <<"HEAD" ;
<?xml version="1.0" encoding ="utf-8"?>
<body
  xmlns    = "http://www.w3.org/1999/xhtml"
  xmlns:kb = "http://herbaer.de/xmlns/20151211/kalenderbilder/"
HEAD
      print $fh "  xmlns:l  = \"http://herbaer.de/xmlns/20141210/localization\"\n"
        if $args -> {"monmode"} eq "i";
      print $fh "  kb:y = \"$year\">\n";
      for ($m = 0; $m < 14; ++$m) {
         print_monat ($args, $data, $m);
      }
      print $fh "</body>\n";
      close $fh;
   }
} # print_kalender
```

```
# Ein Monatsblatt ausgeben
sub print_monat {
   my ($args, $data, $m) = @_;
   my $verbose = $args -> {"[cnt]verbose"};
   my $mon     = $data -> {"monate"} -> [$m]; # Liste zum Monat
   my $year    = $args -> {"year"};
   my $mm      = sprintf ("%02d", $m);
   my $monmode = $args -> {"monmode"};
   my $title;
   my $id;     # ID des Monatnamens in der Lokalisierungstabelle
   if ($monmode eq "num") {
      $title = sprintf ("%02d - %d",
         ($m + 11) % 12 + 1,
         $year + floor (($m + 11) / 12) - 1
      );
   }
   elsif ($monmode eq "i") {
      $id = lc ($mon -> [0]);
      $id =~ s/ä/ae/g;
      $id =~ s/ö/oe/g;
      $id =~ s/ü/ue/g;
      $id =~ s/ß/sz/g;
      $title =
        "<l:ph id = \"$id\"/> "
        . sprintf ("%d", $year + floor (($m + 11) / 12) - 1);
   }
   else {
      $title = sprintf ("%s %d",
         $mon -> [0],
         $year + floor (($m + 11) / 12) - 1
      );
   }
   my $fh = $data -> {"fh"};
   print $fh <<"HEAD" ;
<div id="d$mm">
  <h1>$title</h1>
HEAD
   my $d;            # Laufender Tag
   my $n;            # auszugebende Zahl
   my $nr_cell;      # lfd Numer der Zelle;
   my $class;        # Klasse des laufenden Tags
   my $date;         # Datum im Format mm-tt
   my $numcol;       # Anzahl der Spalten bei senkrechter Ausgabe
   my $row;          # laufende Zeile
   my $col;          # laufende Spalte

   my $aw  = anzwochen ($year);      # Anzahl der Wochen akt. Jahr
   my $awv = anzwochen ($year - 1);  # ... Vorjahr
   my $printweek = sub {
      # erster Tag der ersten Woche
      # (11 - wotagstart) % 7 - 2
      # Woche
      # floor ( (ltag - ((11 - wotagstart) % 7 - 2) + 7) / 7 )
      # = floor ( (ltag - (11 - wotagstart) % 7 + 9) / 7 )
      # = floor ( $d + $mon [2] + 8 - (11 - wotagstart) % 7) / 7)
      my $w = floor ( ($d + $mon -> [2] + 8 - (11 - $data -> {"wotagstart"}) % 7) / 7);
      if    ($w < 1)   { $w += $awv; }
      elsif ($w > $aw) { $w -= $aw;  }
      print $fh "<td class = \"kw\">$w</td>\n";
   };

   my $wd   = $args -> {"[cnt]wd"};
   my $wday;                         # 1 bis 7 für Mo bis So
   my $printdate = sub {
      $class = "";
      if ($d < 1) {
         $n = $d + ($m == 0 ? 30 : $data -> {"monate"} -> [$m - 1] -> [1]);
         $class .= " p";
      }
      elsif ($d > $mon -> [1]) {
         $n = $d - $mon -> [1];
         $class .= " n";
      }
      else {
         $n = $d;
      }
      $date = sprintf ("%02d-%02d", $m, $d);
      $wday = ($mon -> [3] + $d + 5) % 7 + 1;
      $class .= " f"      if $data -> {"f"} -> {$date};
      $class .= " w$wday" if $wd || $wday == 7;
      $class =~ s/^\s+//;
      print $fh "<td";
      print $fh " class=\"$class\"" if $class;
      print $fh ">$n</td>\n";
   };

   my $weekno = $args -> {"[cnt]weekno"};
   my $moncol = $args -> {"[cnt]moncol"};

   my $printmnth = sub {
      print $fh "<td class = \"mc\">";
      if ($monmode eq "num") {
         print $fh sprintf (
            "%02d-%d",
            ($m + 11) % 12 + 1,
            $year + floor (($m + 11) / 12) - 1
```

```
            );
      }
      elsif ($monmode eq "i") {
         $id = lc ($mon -> [0]);
         $id =~ s/ä/ae/g;
         $id =~ s/ö/oe/g;
         $id =~ s/ü/ue/g;
         $id =~ s/ß/sz/g;
         print $fh "<l:ph id = \"$id\"/>";
      }
      else {
         print $fh $mon -> [0];
      }
      print $fh "</td>";
   }; # printmnth

   my $mode;
   my $modkz;  # "h" oder "v"
   my $modno;  # 7 oder 14
   for $mode (@{$args -> {"mode"}}) {
      if ( $mode !~ /^([hv])(7|14)$/ ) {
         print STDERR "ungültiges Tabellenformat --mode \"$mode\"\n"
            if $args -> {"[cnt]verbose"};
         next;
      }
      $modkz = $1;
      $modno = $2;
      if ($modkz eq "h") {
         $nr_cell = 0;
         print $fh "<table class=\"h$modno\">\n";
         for (
            $d = 2 - $mon -> [3];
            $d < 2 - $mon -> [3] + ($mon -> [1] - 1 + $mon -> [3])
            + (42 - ($mon -> [1] - 1 + $mon -> [3])) % $modno ;
            ++$d)
         {
            if ($nr_cell % $modno == 0) {
               print $fh "<tr>\n";
               if ($moncol) {
                  if ($nr_cell == 0) { $printmnth -> (); }
                  else               { print $fh "<td class = \"mc\"/>"; }
               }
               $printweek -> () if $weekno;
            }
            $printdate -> ();
            ++$nr_cell;
            print $fh "</tr>\n" if $nr_cell % $modno == 0;
         }
         print $fh "</table>\n";
      }
      elsif ($modkz eq "v") {
         $nr_cell =
            ($mon -> [1] - 1 + $mon -> [3])
            + (42 - ($mon -> [1] - 1 + $mon -> [3])) % $modno ;
         print $fh "<table class=\"v$modno\">\n";
         if ($moncol) {
             print $fh "<tr class = \"mc\">";
             $printmnth -> ();
             for ($col = 1; $col * $modno < $nr_cell; ++$col) {
                print $fh "<td class = \"mc\"/>";
             }
             print $fh "</tr>\n";
         }
         if ($weekno) {
             print $fh "<tr class = \"kw\">\n";
             for ($col = 0; $col * $modno < $nr_cell; ++$col) {
                 $d = 2 - $mon -> [3] + $col * $modno;
                 $printweek -> ();
             }
         }
         print $fh "</tr>\n";
         for ($row = 0; $row < $modno; ++$row) {
            print $fh "<tr>\n";
            for ($col = 0; $col * $modno < $nr_cell; ++$col) {
               $d = 2 - $mon -> [3] + $col * $modno + $row;
               $printdate -> ();
            }
            print $fh "</tr>\n";
         }
         print $fh "</table>\n";
      }
   }
   print $fh "</div>\n" ;
} # print_monat
```

# Kennnungen der Länder

Die folgende Tabelle enthält die Länder und Gebiete, für die die Website `https://www.timeanddate.de/` Feiertage nennt. Einige Kleinstaaten sind nicht aufgeführt.

| | |
|---|---|
| EG | Ägypten |
| GQ | Äquatorialguinea |
| ET | Äthiopien |
| AF | Afghanistan |
| AL | Albanien |
| GZ | Algerien |
| VI | Amerikanische Jungferninseln |
| AS | Amerikanisch-Samoa |
| AD | Andorra |
| AO | Angola |
| AI | Anguilla |
| AG | Antigua und Barbuda |
| AR | Argentinien |
| AM | Armenien |
| AW | Aruba |
| AZ | Aserbaidschan |
| AU | Australien |
| BS | Bahamas |
| BH | Bahrain |
| BD | Bangladesch |
| BB | Barbados |
| BE | Belgien |
| BZ | Belize |
| BJ | Benin |
| BM | Bermuda |
| BT | Bhutan |
| BO | Bolivien |
| BA | Bosnien und Herzegowina |
| BW | Botswana |
| BR | Brasilien |
| VG | Britische Jungferninseln |
| BN | Brunei Darussalam |
| BG | Bulgarien |
| BF | Burkina Faso |
| BI | Burundi |
| KY | Kaimaninseln |
| CL | Chile |
| CN | Volksrepublik China |
| CK | Cookinseln |

| | |
|---|---|
| CR | Costa Rica |
| CW | Curaçao |
| DK | Dänemark |
| DE | Deutschland |
| DM | Dominica |
| DO | Dominikanische Republik |
| CD | Demokratische Republik Kongo |
| DJ | Dschibuti |
| EC | Ecuador |
| CI | Elfenbeinküste |
| SV | El Salvador |
| ER | Eritrea |
| EE | Estland |
| FK | Falklandinseln |
| FO | Färöer |
| FJ | Fidschi |
| FI | Finnland |
| FR | Frankreich |
| GF | Französisch-Guayana |
| PF | Französisch-Polynesien |
| GA | Gabun |
| GM | Gambia |
| GE | Georgien |
| GH | Ghana |
| GI | Gibraltar |
| GD | Grenada |
| GR | Griechenland |
| GL | Grönland |
| GB | Vereinigtes Königreich Großbritannien und Nordirland |
| GP | Guadeloupe |
| GU | Guam |
| GT | Guatemala |
| GG | Guernsey |
| GN | Guinea |
| GW | Guinea-Bissau |
| GY | Guyana |
| HT | Haiti |
| HN | Honduras |
| HK | Hongkong |
| IN | Indien |
| IM | Insel Man |
| ID | Indonesien |
| IQ | Irak |

| | |
|---|---|
| IR | Iran |
| IE | Irland |
| IS | Island |
| IL | Israel |
| IT | Italien |
| JM | Jamaika |
| JP | Japan |
| YE | Jemen |
| JE | Jersey |
| JO | Jordanien |
| KH | Kambodscha |
| CM | Kamerun |
| CA | Kanada |
| CV | Kap Verde |
| KZ | Kasachstan |
| KE | Kenia |
| KG | Kirgisistan |
| KI | Kiribati |
| CO | Kolumbien |
| KM | Komoren |
| XK | Kosovo |
| HR | Kroatien |
| CU | Kuba |
| KW | Kuwait |
| LA | Laos |
| LS | Lesotho |
| LV | Lettland |
| LB | Libanon |
| LR | Liberia |
| LY | Libyen |
| LI | Liechtenstein |
| LT | Litauen |
| LU | Luxemburg |
| MO | Macau |
| MG | Madagaskar |
| MW | Malawi |
| MY | Malaysia |
| MV | Malediven |
| ML | Mali |
| MT | Malta |
| MA | Marokko |
| MH | Marshallinseln |
| MQ | Martinique |

| | |
|---|---|
| MR | Mauretanien |
| MU | Mauritius |
| YT | Mayotte |
| MK | Mazedonien |
| MX | Mexiko |
| FM | Mikronesien |
| MD | Moldawien |
| MC | Monaco |
| MN | Mongolei |
| ME | Montenegro |
| MS | Montserrat |
| MZ | Mosambik |
| MM | Myanmar |
| NA | Namibia |
| NR | Nauru |
| NP | Nepal |
| NC | Neukaledonien |
| NZ | Neuseeland |
| NI | Nicaragua |
| NL | Niederlande |
| NE | Niger |
| NG | Nigeria |
| MP | Nördliche Marianen |
| KP | Demokratische Volksrepublik Korea |
| NO | Norwegen |
| AT | Österreich |
| OM | Oman |
| TL | Osttimor |
| PK | Pakistan |
| PW | Palau |
| PA | Panama |
| PG | Papua-Neuguinea |
| PY | Paraguay |
| PE | Peru |
| PH | Philippinen |
| PL | Polen |
| PT | Portugal |
| PR | Puerto Rico |
| QA | Katar |
| CG | Republik Kongo |
| RE | Réunion |
| RW | Ruanda |
| RO | Rumänien |

| | |
|---|---|
| RU | Russische Föderation |
| BL | Saint-Barthélemy |
| MF | Saint-Martin |
| PM | Saint-Pierre und Miquelon |
| SB | Salomonen |
| ZM | Sambia |
| WS | Samoa |
| SM | San Marino |
| ST | São Tomé und Príncipe |
| SA | Saudi-Arabien |
| SE | Schweden |
| CH | Schweiz |
| SN | Senegal |
| RS | Serbien |
| SC | Seychellen |
| SL | Sierra Leone |
| ZW | Simbabwe |
| SG | Singapur |
| SX | Sint Maarten |
| SK | Slowakei |
| SI | Slowenien |
| SO | Somalia |
| ES | Spanien |
| LK | Sri Lanka |
| SH | St. Helena |
| KN | St. Kitts und Nevis |
| LC | St. Lucia |
| VC | St. Vincent und die Grenadinen |
| SD | Sudan |
| ZA | Südafrika |
| KR | Republik Korea |
| SS | Südsudan |
| SR | Suriname |
| SZ | Swasiland |
| SY | Syrien |
| TJ | Tadschikistan |
| TW | Taiwan |
| TZ | Tansania |
| TH | Thailand |
| TG | Togo |
| TO | Tonga |
| TT | Trinidad und Tobago |
| TD | Tschad |

| | |
|---|---|
| CZ | Tschechien |
| TR | Türkei |
| TN | Tunesien |
| TM | Turkmenistan |
| TC | Turks- und Caicosinseln |
| TV | Tuvalu |
| UG | Uganda |
| UA | Ukraine |
| HU | Ungarn |
| UY | Uruguay |
| US | Vereinigte Staaten von Amerika |
| UZ | Usbekistan |
| VU | Vanuatu |
| VA | Vatikanstadt |
| VE | Venezuela |
| AE | Vereinigte Arabische Emirate |
| VN | Vietnam |
| WF | Wallis und Futuna |
| BY | Weißrussland |
| CF | Zentralafrikanische Republik |
| CY | Zypern |

Die folgende Tabelle nennt einige Gebiete, für die es einen eigenen Code gibt, die aber zu einem größeren Land gehören (hinsichtlich der Feiertage).

| | |
|---|---|
| TF | Französische Süd- und Antarktisgebiete |
| HM | Heard und McDonaldinseln |

# Sprachencodes

[Quelltext]

Viele Sprachen werden in vielen Ländern gesprochen. Umgekehrt werden auch in vielen Ländern viele Sprachen gesprochen. Für die Kalender ist die Zuordnung einer Sprache zu einem Land nützlich, wenn das Land nicht auf andere Weise angegeben ist. Die folgende Tabelle enthält die Sprachen, die „Google Translate” anbietet. Die Zuordnung einer Sprache zu einem Land ist in einigen Fällen willkürlich.

Ich habe hier die Länderkennungen großgeschrieben. Die Sprachenpräferenzliste (`Accept-Language`) sollte nur Kleinbuchstaben (keine Großbuchstaben) enthalten.

| | |
|---|---|
| 1. Spalte | Sprachencode |
| 2. Spalte | Sprache |
| 3. Spalte | Ländercode |
| 4. Spalte | Land |

| | | | |
|---|---|---|---|
| af | Afrikaans | ZA | Südafrika |
| am | Amharisch | ET | Äthiopien |
| ar | Arabisch | EG | Ägypten |
| az | Aserbaidschanisch | AZ | Aserbaidschan |
| be | Belorussisch | BY | Weißrussland |
| bg | Bulgarisch | BG | Bulgarien |
| bn | Bengalisch | BD | Bangladesch |
| bs | Bosnisch | BA | Bosnien und Herzegowina |
| ca | Katalanisch | ES | Spanien |
| ceb | Cebuano | PH | Philippinen |
| co | Korsisch | FR | Frankreich |
| cs | Tschechisch | CZ | Tschechien |
| cy | Walisisch | UK | Vereinigtes Königreich |
| da | Dänisch | DK | Dänemark |
| de | Deutsch | DE | Deutschland |
| el | Griechisch | GR | Griechenland |
| en | Englisch | UK | Großbritannien |
| eo | Esperanto | CN | China |
| es | Spanisch | ES | Spanien |
| et | Estnisch | EE | Estland |
| eu | Baskisch | ES | Spanien |
| fa | Persisch | IR | Iran |
| fi | Finnisch | FI | Finnland |
| fr | Französisch | FR | Frankreich |
| fy | Friesisch | NL | Niederlande |
| ga | Irisch | IE | Irland |
| gd | Schottisches Gälisch | UK | Großbritannien |
| gl | Galizisch | ES | Spanien |
| gu | Gujaratisch | IN | Indien |

| | | | |
|---|---|---|---|
| ha | Hausa | NG | Nigeria |
| haw | Hawaiisch | US | Vereinigte Staaten |
| hi | Hindi | IN | Indien |
| hmn | Hmong | CN | China |
| hr | Kroatisch | HR | Kroatien |
| ht | Haitianisch | HT | Haiti |
| hu | Ungarisch | HU | Ungarn |
| hy | Armenisch | AM | Armenien |
| id | Indonesisch | MY | Malaysia |
| ig | Igbo | NG | Nigeria |
| is | Isländisch | IS | Island |
| it | Italienisch | IT | Italien |
| iw | Hebräisch | IL | Israel |
| ja | Japanisch | JP | Japa |
| jw | Javanisch | ID | Indonesien |
| ka | Georgisch | GE | Georgien |
| kk | Kasachisch | KZ | Kasachstan |
| km | Kambodschanisch | KH | Kambodscha |
| kn | Kannada | IN | Indien |
| ko | Koreanisch | KR | Korea |
| ku | Kurdisch | IQ | Irak |
| ky | Kirgisisch | KG | Kirgisistan |
| la | Lateinisch | VA | Vatikanstadt |
| lb | Luxemburgisch | LU | Luxemburg |
| lo | Laotisch | LA | Laos |
| lt | Litauisch | LT | Litauen |
| lv | Lettisch | LV | Lettland |
| mg | Madagassisch | MG | Madagaskar |
| mi | Maorisch | NZ | Neuseeland |
| mk | Mazedonisch | MK | Mazedonien |
| ml | Malajalam | IN | Indien |
| mn | Mongolisch | MN | Mongolei |
| mr | Marathi | IN | Indien |
| ms | Malaysisch | MY | Malaysia |
| mt | Maltesisch | MT | Malta |
| my | Burmesisch | MM | Myanmar |
| ne | Nepalesisch | NP | Nepal |
| nl | Holländisch | NL | Niederlande |
| no | Norwegisch | NO | Norwegen |
| ny | Chichewa | MW | Malawi |
| pa | Panjabi | PK | Pakistan |
| pl | Polnisch | PL | Polen |
| ps | Paschtu | AF | Afghanistan |

| | | | |
|---|---|---|---|
| pt | Portugiesisch | PT | Portugal |
| ro | Rumänisch | RO | Rumänien |
| ru | Russisch | RU | Russische Föderation |
| sd | Sindhi | PK | Pakistan |
| si | Singhalesisch | LK | Sri Lanka |
| sk | Slowakisch | SK | Slowakei |
| sl | Slowenisch | SI | Slowenien |
| sm | Samoanisch | WS | Samoa |
| sn | Schona | ZW | Simbabwe |
| so | Somali | SO | Somalia |
| sq | Albanisch | AL | Albanien |
| sr | Serbisch | RS | Serbien |
| st | Sesotho | LS | Lesotho |
| su | Sundanesisch | ID | Indonesien |
| sv | Schwedisch | SE | Schweden |
| sw | Suaheli | TZ | Tansania |
| ta | Tamil | IN | Indien |
| te | Telugu | IN | Indien |
| tg | Tadschikisch | TJ | Tadschikistan |
| th | Thai | TH | Thailand |
| tl | Tagalog | PH | Philippinen |
| tr | Türkisch | TR | Türkei |
| uk | Ukrainisch | UA | Ukraine |
| ur | Urdu | PK | Pakistan |
| uz | Usbekisch | UZ | Usbekistan |
| vi | Vietnamesisch | VN | Vietnam |
| xh | Xhosa | ZA | Südafrika |
| yi | Jiddish | PL | Polen |
| yo | Joruba | BJ | Benin |
| zh | Chinesisch | CN | China |
| zu | Zulu | ZA | Südafrika |

# Quelltext

[Beschreibung]

```
<?xml version="1.0" encoding="utf-8"?>
<!--
  Sprachen-Codes
  Herbert Schiemann <h.schiemann@herbaer.de>
  Borkener Str. 167, 46284 Dorsten, Germany
  GPL Version 2 oder neuer
  Jede Gewährleistung ist ausgeschlossen.
-->
<article
    xmlns = "http://docbook.org/ns/docbook"
    xml:lang = "de"
    xml:id   = "langcodes"
    version="5.0"
>
<info>
  <title>Sprachencodes</title>
  <date>2017-05-27</date>
  <author>
    <personname>
      <firstname>Herbert</firstname>
      <surname>Schiemann</surname>
    </personname>
    <email>h.schiemann@herbaer.de</email>
  </author>
</info>


<para>
Viele Sprachen werden in vielen Ländern gesprochen.
Umgekehrt werden auch in vielen Ländern viele Sprachen gesprochen.
Für die Kalender ist die Zuordnung einer Sprache zu einem Land nützlich,
wenn das Land nicht auf andere Weise angegeben ist.
Die folgende Tabelle enthält die Sprachen,
die &#x201e;Google Translate&#x201d; anbietet.
Die Zuordnung einer Sprache zu einem Land ist in einigen Fällen willkürlich.
</para>
<para>
Ich habe hier die Länderkennungen großgeschrieben.
Die Sprachenpräferenzliste (<literal>Accept-Language</literal>)
sollte nur Kleinbuchstaben (keine Großbuchstaben) enthalten.
</para>
<informaltable frame="none" rowsep="0" colsep="0">
  <tgroup cols="2">
    <tbody>
      <row>
        <entry>1. Spalte</entry>
        <entry>Sprachencode</entry>
      </row>
      <row>
        <entry>2. Spalte</entry>
        <entry>Sprache</entry>
      </row>
      <row>
        <entry>3. Spalte</entry>
        <entry>Ländercode</entry>
      </row>
      <row>
        <entry>4. Spalte</entry>
        <entry>Land</entry>
      </row>
    </tbody>
  </tgroup>
</informaltable>


<informaltable frame="none" rowsep="0" colsep="0" xml:id = "tab_langcodes">
  <tgroup cols="4">
    <tbody>
<row>
<entry><literal>af</literal></entry><entry>Afrikaans</entry>
<entry><literal>ZA</literal></entry><entry>Südafrika</entry>


</row>
<row>
<entry><literal>am</literal></entry><entry>Amharisch</entry>
<entry><literal>ET</literal></entry><entry>Äthiopien</entry>


</row>
<row>
<entry><literal>ar</literal></entry><entry>Arabisch</entry>
<entry><literal>EG</literal></entry><entry>Ägypten</entry>


</row>
<row>
<entry><literal>az</literal></entry><entry>Aserbaidschanisch</entry>
<entry><literal>AZ</literal></entry><entry>Aserbaidschan</entry>
```

```
</row>
<row>
<entry><literal>be</literal></entry><entry>Belorussisch</entry>
<entry><literal>BY</literal></entry><entry>Weißrussland</entry>

</row>
<row>
<entry><literal>bg</literal></entry><entry>Bulgarisch</entry>
<entry><literal>BG</literal></entry><entry>Bulgarien</entry>

</row>
<row>
<entry><literal>bn</literal></entry><entry>Bengalisch</entry>
<entry><literal>BD</literal></entry><entry>Bangladesch</entry>

</row>
<row>
<entry><literal>bs</literal></entry><entry>Bosnisch</entry>
<entry><literal>BA</literal></entry><entry>Bosnien und Herzegowina</entry>

</row>
<row>
<entry><literal>ca</literal></entry><entry>Katalanisch</entry>
<entry><literal>ES</literal></entry><entry>Spanien</entry>

</row>
<row>
<entry><literal>ceb</literal></entry><entry>Cebuano</entry>
<entry><literal>PH</literal></entry><entry>Philippinen</entry>
</row>
<row>
<entry><literal>co</literal></entry><entry>Korsisch</entry>
<entry><literal>FR</literal></entry><entry>Frankreich</entry>

</row>
<row>
<entry><literal>cs</literal></entry><entry>Tschechisch</entry>
<entry><literal>CZ</literal></entry><entry>Tschechien</entry>

</row>
<row>
<entry><literal>cy</literal></entry><entry>Walisisch</entry>
<entry><literal>UK</literal></entry><entry>Vereinigtes Königreich</entry>

</row>
<row>
<entry><literal>da</literal></entry><entry>Dänisch</entry>
<entry><literal>DK</literal></entry><entry>Dänemark</entry>

</row>
<row>
<entry><literal>de</literal></entry><entry>Deutsch</entry>
<entry><literal>DE</literal></entry><entry>Deutschland</entry>

</row>
<row>
<entry><literal>el</literal></entry><entry>Griechisch</entry>
<entry><literal>GR</literal></entry><entry>Griechenland</entry>

</row>
<row>
<entry><literal>en</literal></entry><entry>Englisch</entry>
<entry><literal>UK</literal></entry><entry>Großbritannien</entry>

</row>
<row>
<entry><literal>eo</literal></entry><entry>Esperanto</entry>
<entry><literal>CN</literal></entry><entry>China</entry>
</row>
<row>
<entry><literal>es</literal></entry><entry>Spanisch</entry>
<entry><literal>ES</literal></entry><entry>Spanien</entry>

</row>
<row>
<entry><literal>et</literal></entry><entry>Estnisch</entry>
<entry><literal>EE</literal></entry><entry>Estland</entry>

</row>
<row>
<entry><literal>eu</literal></entry><entry>Baskisch</entry>
<entry><literal>ES</literal></entry><entry>Spanien</entry>
```

```
</row>
<row>
<entry><literal>fa</literal></entry><entry>Persisch</entry>
<entry><literal>IR</literal></entry><entry>Iran</entry>


</row>
<row>
<entry><literal>fi</literal></entry><entry>Finnisch</entry>
<entry><literal>FI</literal></entry><entry>Finnland</entry>


</row>
<row>
<entry><literal>fr</literal></entry><entry>Französisch</entry>
<entry><literal>FR</literal></entry><entry>Frankreich</entry>


</row>
<row>
<entry><literal>fy</literal></entry><entry>Friesisch</entry>
<entry><literal>NL</literal></entry><entry>Niederlande</entry>


</row>
<row>
<entry><literal>ga</literal></entry><entry>Irisch</entry>
<entry><literal>IE</literal></entry><entry>Irland</entry>


</row>
<row>
<entry><literal>gd</literal></entry><entry>Schottisches Gälisch</entry>
<entry><literal>UK</literal></entry><entry>Großbritannien</entry>


</row>
<row>
<entry><literal>gl</literal></entry><entry>Galizisch</entry>
<entry><literal>ES</literal></entry><entry>Spanien</entry>


</row>
<row>
<entry><literal>gu</literal></entry><entry>Gujaratisch</entry>
<entry><literal>IN</literal></entry><entry>Indien</entry>


</row>
<row>
<entry><literal>ha</literal></entry><entry>Hausa</entry>
<entry><literal>NG</literal></entry><entry>Nigeria</entry>


</row>
<row>
<entry><literal>haw</literal></entry><entry>Hawaiisch</entry>
<entry><literal>US</literal></entry><entry>Vereinigte Staaten</entry>


</row>
<row>
<entry><literal>hi</literal></entry><entry>Hindi</entry>
<entry><literal>IN</literal></entry><entry>Indien</entry>


</row>
<row>
<entry><literal>hmn</literal></entry><entry>Hmong</entry>
<entry><literal>CN</literal></entry><entry>China</entry>


</row>
<row>
<entry><literal>hr</literal></entry><entry>Kroatisch</entry>
<entry><literal>HR</literal></entry><entry>Kroatien</entry>


</row>
<row>
<entry><literal>ht</literal></entry><entry>Haitianisch</entry>
<entry><literal>HT</literal></entry><entry>Haiti</entry>
</row>
<row>
<entry><literal>hu</literal></entry><entry>Ungarisch</entry>
<entry><literal>HU</literal></entry><entry>Ungarn</entry>


</row>
<row>
<entry><literal>hy</literal></entry><entry>Armenisch</entry>
<entry><literal>AM</literal></entry><entry>Armenien</entry>
```

```
</row>
<row>
<entry><literal>id</literal></entry><entry>Indonesisch</entry>
<entry><literal>MY</literal></entry><entry>Malaysia</entry>


</row>
<row>
<entry><literal>ig</literal></entry><entry>Igbo</entry>
<entry><literal>NG</literal></entry><entry>Nigeria</entry>


</row>
<row>
<entry><literal>is</literal></entry><entry>Isländisch</entry>
<entry><literal>IS</literal></entry><entry>Island</entry>


</row>
<row>
<entry><literal>it</literal></entry><entry>Italienisch</entry>
<entry><literal>IT</literal></entry><entry>Italien</entry>


</row>
<row>
<entry><literal>iw</literal></entry><entry>Hebräisch</entry>
<entry><literal>IL</literal></entry><entry>Israel</entry>


</row>
<row>
<entry><literal>ja</literal></entry><entry>Japanisch</entry>
<entry><literal>JP</literal></entry><entry>Japa</entry>


</row>


<row>
<entry><literal>jw</literal></entry><entry>Javanisch</entry>
<entry><literal>ID</literal></entry><entry>Indonesien</entry>


</row>
<row>
<entry><literal>ka</literal></entry><entry>Georgisch</entry>
<entry><literal>GE</literal></entry><entry>Georgien</entry>


</row>
<row>
<entry><literal>kk</literal></entry><entry>Kasachisch</entry>
<entry><literal>KZ</literal></entry><entry>Kasachstan</entry>


</row>
<row>
<entry><literal>km</literal></entry><entry>Kambodschanisch</entry>
<entry><literal>KH</literal></entry><entry>Kambodscha</entry>


</row>
<row>
<entry><literal>kn</literal></entry><entry>Kannada</entry>
<entry><literal>IN</literal></entry><entry>Indien</entry>


</row>
<row>
<entry><literal>ko</literal></entry><entry>Koreanisch</entry>
<entry><literal>KR</literal></entry><entry>Korea</entry>


</row>
<row>
<entry><literal>ku</literal></entry><entry>Kurdisch</entry>
<entry><literal>IQ</literal></entry><entry>Irak</entry>


</row>
<row>
<entry><literal>ky</literal></entry><entry>Kirgisisch</entry>
<entry><literal>KG</literal></entry><entry>Kirgisistan</entry>
```

```
</row>
<row>
<entry><literal>la</literal></entry><entry>Lateinisch</entry>
<entry><literal>VA</literal></entry><entry>Vatikanstadt</entry>
</row>
<row>
<entry><literal>lb</literal></entry><entry>Luxemburgisch</entry>
<entry><literal>LU</literal></entry><entry>Luxemburg</entry>
</row>
<row>
<entry><literal>lo</literal></entry><entry>Laotisch</entry>
<entry><literal>LA</literal></entry><entry>Laos</entry>


</row>
<row>
<entry><literal>lt</literal></entry><entry>Litauisch</entry>
<entry><literal>LT</literal></entry><entry>Litauen</entry>


</row>
<row>
<entry><literal>lv</literal></entry><entry>Lettisch</entry>
<entry><literal>LV</literal></entry><entry>Lettland</entry>


</row>
<row>
<entry><literal>mg</literal></entry><entry>Madagassisch</entry>
<entry><literal>MG</literal></entry><entry>Madagaskar</entry>


</row>
<row>
<entry><literal>mi</literal></entry><entry>Maorisch</entry>
<entry><literal>NZ</literal></entry><entry>Neuseeland</entry>
</row>
<row>
<entry><literal>mk</literal></entry><entry>Mazedonisch</entry>
<entry><literal>MK</literal></entry><entry>Mazedonien</entry>


</row>
<row>
<entry><literal>ml</literal></entry><entry>Malajalam</entry>
<entry><literal>IN</literal></entry><entry>Indien</entry>


</row>
<row>
<entry><literal>mn</literal></entry><entry>Mongolisch</entry>
<entry><literal>MN</literal></entry><entry>Mongolei</entry>


</row>
<row>
<entry><literal>mr</literal></entry><entry>Marathi</entry>
<entry><literal>IN</literal></entry><entry>Indien</entry>


</row>
<row>
<entry><literal>ms</literal></entry><entry>Malaysisch</entry>
<entry><literal>MY</literal></entry><entry>Malaysia</entry>
</row>
<row>
<entry><literal>mt</literal></entry><entry>Maltesisch</entry>
<entry><literal>MT</literal></entry><entry>Malta</entry>


</row>
<row>
<entry><literal>my</literal></entry><entry>Burmesisch</entry>
<entry><literal>MM</literal></entry><entry>Myanmar</entry>


</row>
<row>
<entry><literal>ne</literal></entry><entry>Nepalesisch</entry>
<entry><literal>NP</literal></entry><entry>Nepal</entry>


</row>
<row>
<entry><literal>nl</literal></entry><entry>Holländisch</entry>
<entry><literal>NL</literal></entry><entry>Niederlande</entry>


</row>
<row>
<entry><literal>no</literal></entry><entry>Norwegisch</entry>
<entry><literal>NO</literal></entry><entry>Norwegen</entry>


</row>
<row>
<entry><literal>ny</literal></entry><entry>Chichewa</entry>
<entry><literal>MW</literal></entry><entry>Malawi</entry>


</row>
```

```
<row>
<entry><literal>pa</literal></entry><entry>Panjabi</entry>
<entry><literal>PK</literal></entry><entry>Pakistan</entry>

</row>
<row>
<entry><literal>pl</literal></entry><entry>Polnisch</entry>
<entry><literal>PL</literal></entry><entry>Polen</entry>

</row>
<row>
<entry><literal>ps</literal></entry><entry>Paschtu</entry>
<entry><literal>AF</literal></entry><entry>Afghanistan</entry>

</row>
<row>
<entry><literal>pt</literal></entry><entry>Portugiesisch</entry>
<entry><literal>PT</literal></entry><entry>Portugal</entry>

</row>
<row>
<entry><literal>ro</literal></entry><entry>Rumänisch</entry>
<entry><literal>RO</literal></entry><entry>Rumänien</entry>

</row>
<row>
<entry><literal>ru</literal></entry><entry>Russisch</entry>
<entry><literal>RU</literal></entry><entry>Russische Föderation</entry>

</row>
<row>
<entry><literal>sd</literal></entry><entry>Sindhi</entry>
<entry><literal>PK</literal></entry><entry>Pakistan</entry>

</row>
<row>
<entry><literal>si</literal></entry><entry>Singhalesisch</entry>
<entry><literal>LK</literal></entry><entry>Sri Lanka</entry>
</row>
<row>
<entry><literal>sk</literal></entry><entry>Slowakisch</entry>
<entry><literal>SK</literal></entry><entry>Slowakei</entry>

</row>
<row>
<entry><literal>sl</literal></entry><entry>Slowenisch</entry>
<entry><literal>SI</literal></entry><entry>Slowenien</entry>

</row>
<row>
<entry><literal>sm</literal></entry><entry>Samoanisch</entry>
<entry><literal>WS</literal></entry><entry>Samoa</entry>

</row>
<row>
<entry><literal>sn</literal></entry><entry>Schona</entry>
<entry><literal>ZW</literal></entry><entry>Simbabwe</entry>

</row>
<row>
<entry><literal>so</literal></entry><entry>Somali</entry>
<entry><literal>SO</literal></entry><entry>Somalia</entry>

</row>
<row>
<entry><literal>sq</literal></entry><entry>Albanisch</entry>
<entry><literal>AL</literal></entry><entry>Albanien</entry>

</row>
<row>
<entry><literal>sr</literal></entry><entry>Serbisch</entry>
<entry><literal>RS</literal></entry><entry>Serbien</entry>

</row>
<row>
<entry><literal>st</literal></entry><entry>Sesotho</entry>
<entry><literal>LS</literal></entry><entry>Lesotho</entry>

</row>
<row>
<entry><literal>su</literal></entry><entry>Sundanesisch</entry>
<entry><literal>ID</literal></entry><entry>Indonesien</entry>

</row>
<row>
<entry><literal>sv</literal></entry><entry>Schwedisch</entry>
```

```
<entry><literal>SE</literal></entry><entry>Schweden</entry>

</row>
<row>
<entry><literal>sw</literal></entry><entry>Suaheli</entry>
<entry><literal>TZ</literal></entry><entry>Tansania</entry>

</row>
<row>
<entry><literal>ta</literal></entry><entry>Tamil</entry>
<entry><literal>IN</literal></entry><entry>Indien</entry>

</row>
<row>
<entry><literal>te</literal></entry><entry>Telugu</entry>
<entry><literal>IN</literal></entry><entry>Indien</entry>

</row>
<row>
<entry><literal>tg</literal></entry><entry>Tadschikisch</entry>
<entry><literal>TJ</literal></entry><entry>Tadschikistan</entry>

</row>
<row>
<entry><literal>th</literal></entry><entry>Thai</entry>
<entry><literal>TH</literal></entry><entry>Thailand</entry>
</row>
<row>
<entry><literal>tl</literal></entry><entry>Tagalog</entry>
<entry><literal>PH</literal></entry><entry>Philippinen</entry>

</row>
<row>
<entry><literal>tr</literal></entry><entry>Türkisch</entry>
<entry><literal>TR</literal></entry><entry>Türkei</entry>

</row>
<row>
<entry><literal>uk</literal></entry><entry>Ukrainisch</entry>
<entry><literal>UA</literal></entry><entry>Ukraine</entry>

</row>
<row>
<entry><literal>ur</literal></entry><entry>Urdu</entry>
<entry><literal>PK</literal></entry><entry>Pakistan</entry>

</row>
<row>
<entry><literal>uz</literal></entry><entry>Usbekisch</entry>
<entry><literal>UZ</literal></entry><entry>Usbekistan</entry>

</row>
<row>
<entry><literal>vi</literal></entry><entry>Vietnamesisch</entry>
<entry><literal>VN</literal></entry><entry>Vietnam</entry>

</row>
<row>
<entry><literal>xh</literal></entry><entry>Xhosa</entry>
<entry><literal>ZA</literal></entry><entry>Südafrika</entry>

</row>
<row>
<entry><literal>yi</literal></entry><entry>Jiddish</entry>
<entry><literal>PL</literal></entry><entry>Polen</entry>

</row>
<row>
<entry><literal>yo</literal></entry><entry>Joruba</entry>
<entry><literal>BJ</literal></entry><entry>Benin</entry>

</row>
<row>
<entry><literal>zh</literal></entry><entry>Chinesisch</entry>
<entry><literal>CN</literal></entry><entry>China</entry>

</row>
<row>
<entry><literal>zu</literal></entry><entry>Zulu</entry>
<entry><literal>ZA</literal></entry><entry>Südafrika</entry>

</row>
</tbody>
</tgroup>
</informaltable>

</article>
```

# langcodes_cmd.xslt

[Quelltext]

## Namensräume

| Präfix | Namensraum |
|---|---|
| xml | http://www.w3.org/XML/1998/namespace |
| db | http://docbook.org/ns/docbook |
| d | http://herbaer.de/xmlns/20051201/doc |
| xsl | http://www.w3.org/1999/XSL/Transform |

## Ausgabe (output)

| | |
|---|---|
| Method | text |
| Encoding | utf-8 |

## Eingebundene Stylesheets

### /pool/txt.xslt - Hilfsvorlagen zur Ausgabe und Verarbeitung von Text

Zeilenumbruch txt.break

## Parameter

### Parameter g_cmd

Der Befehl für jede Sprache

Select: 'proc_langbase'

Der Parameter wird in den folgenden Toplevel-Elementen benutzt:

Muster-Vorlage db:row

### Parameter g_tabid

Die ID der Tabelle

Select: 'tab_langcodes'

Der Parameter wird in den folgenden Toplevel-Elementen benutzt:

Muster-Vorlage /

## Muster-Vorlagen (matching templates)

### Muster-Vorlage /

Wurzel

Verwendete globale Parameter oder Variable:

Parameter g_tabid

## Muster-Vorlage db:row

Ein Befehl für jeder Tabellenzeile

Verwendete globale Parameter oder Variable:

Parameter g_cmd

# Quelltext

[Beschreibung]

```
<?xml version="1.0" encoding="utf-8"?>
<?xml-stylesheet href="xslt_ht.xslt" type="application/xml"?>
<!--
2020-12-17 Länderkennung in Großbuchstaben
-->
<xsl:stylesheet
  xmlns:xsl = "http://www.w3.org/1999/XSL/Transform"
  xmlns:d   = "http://herbaer.de/xmlns/20051201/doc"
  xmlns:db  = "http://docbook.org/ns/docbook"
  version   = "1.0"
>


<xsl:param name = "g_cmd" select = "'proc_langbase'"/>


<xsl:param name = "g_tabid" select = "'tab_langcodes'"/>


<xsl:include href = "/pool/txt.xslt"/>


<xsl:output method = "text" encoding = "utf-8"/>


<xsl:template match = "/">
  <xsl:apply-templates select = "//db:informaltable [@xml:id = $g_tabid]//db:row"/>
</xsl:template>


<xsl:template match = "db:row">
  <xsl:variable name = "lang" select = "(.//db:literal)[1]"/>
  <xsl:variable name = "country" select = "(.//db:literal)[2]"/>
  <xsl:value-of select = "concat (
      $g_cmd, ' ', $lang, ' ',
      translate ($country, 'abcdefghijklmnopqrstuvwxyz', 'ABCDEFGHIJKLMNOPQRSTUVWXYZ'),
      ';',
      $txt.break
    )"/>
</xsl:template>


</xsl:stylesheet>
```

# Darstellung der Kalender

Die URI der Kalender sind die URI der Bildauswahldateien

Die Transformation `kal.xslt` bestimmt die Darstellung der Kalender. Vorlagen für die Bildauswahldateien (XML-Namensraum `http://herbaer.de/xmlns/20151211/kalenderbilder/` [`kalenderbilder.rng`]) `http://kleider.herbaer.de/kal/`*`JAHR/BILDAUSWAHL`*`.xml` stellen die Daten (Titel und Bildverweise) zu einer Zeichenkette zusammen. Diese Zeichenkette führt zusammen mit der Grunddaten-Datei zur XHTML-Darstellung, die zum Ausdrucken und zur Ansicht im Browser geeignet ist.

Die Kalenderansicht der Bildergeschichten (`style/kal.xslt`) stellt Titel und Bildverweise aus der Bildergeschichte zusammen und bindet diese Transformation (`kal.xslt`) ein.

Die Darstellung wird ergänzt durch die CSS-Regeln (`kal.css`) und Javascript (`kal.js`) für die Bildschirmansicht und CSS-Regeln für die Druckausgabe (`prt.css`). Die Quelldateien werden zum Dokument unter dem URI `/kal/s/kal.xslt` zusammengefügt. Die sprachabhängigen Texte stehen in der zentralen Lokalisierungsdatei unter dem URI `/local/local.xml`.

Zur Ansicht der Kalender gehört die Hilfe `kal_help.xhtml.de` zusammen mit den CSS-Regeln `kal_help.css` und dem Skript `kal_help.js`. Diese Quelldateien werden zu einem Dokument unter dem URI `/kal/s/kal_help.xhtml` zusammengefügt und in verschiedene Sprachen übersetzt.

# kal.css

[Quelltext]

## Die Rolle dieser Datei

Diese Datei wird von der Transformation `kal.xslt` eingebunden. Sie ist für eine Ansicht vorgesehen, die die Tage einer Woche in einer waagerechten Zeile anzeigt.

## Einzelheiten zu den Regeln

Gesamtgröße (`body`)

> Das `body`-Element wird absolut mit einer festen Breite positioniert. Eine Anpassung an die Breite des Fensters sieht nicht schön aus.

Deckblatt und Monatsblätter

> Die CSS-Regeln verbergen zunächst alle `div`-Elemente, die Kindelemente des `body`-Elements sind. Javascript-Funktionen zeigen immer genau ein „Blatt” an.

Tabelle der Tage

> In den Tabellen der Tage werden der Vormonat, Folgemonat, Tage des Vormonats und des Folgemonats, Feiertage und Sonntage farblich gekennzeichnet. Die Tabellen der verschiedenen Monate sollten gleiche Spaltenbreiten haben, wenn die Übersicht des Vormonats oder des Folgemonats angezeigt wird.

Randbereiche

> Die „Randbereiche” sind der rechte, der linke und der obere Rand des `body`-Elements. Die fest posistionierten Randbereiche sind in der Regel nicht sichtbar (transparent), nur wenn der Zeiger auf einen Randbereich weist, wird der Inhalt schwarz dargestellt.

Verborgene Elemente

> Elemente, deren Attibutwert `class` das Wort `hide` enthält, werden nicht angezeigt. Außer dem Fenster „Einstellung” können das die Übersicht des Vormonats, des Folgemonats oder die Tabellenspalten mit dem Monatsnamen oder der Nummer der Kalenderwoche sein, gesteuert durch Javascript.

# Quelltext

[Beschreibung]

```
/* 2015-12-13 Herbert Schiemann <h.schiemann@herbaer.de> */

body {
    position: absolute;
    width: 30em;
}

div { text-align: center;}

/* Deckblatt und Monatsblatt */
body > div { display: none; }

/* "Kachel"-Bilder auf dem Deckblatt */
img[class~="d"] {
    width: 20%;
    margin: 2% 2%;
}

/* Überschriften des Deckblatts, der Monatsblätter und des Fenster "Einstellung" */
h1, h3 {
    font-family: sans-serif;
}

/* Bild auf einem Monatsblatt */
img[class~="m"] {
    width: 80%;
}

/* Tabelle der Kalendertage */
table {
    width: 95%;
    margin: 2%;
}

/* Übersicht des Vormonats */
table[class~="v"] { background-color: #EEDDDD; }

/* Übersicht des Folgemonats */
table[class~="n"] { background-color: #DDEEDD; }

td[class~="kw"] { font-size: 80%;   width: 2em;} /* Kalenderwoche         */
td[class~="mc"] { text-align: left; width: 6em;} /* Monatsname in Tabelle */
td[class~="f"]  { color: #FF0000; }              /* Feiertag              */
td[class~="w7"] { color: #FF0000; }              /* Sonntag               */
td[class~="p"]  { color: #AAAAAA; }              /* Tag des Vormonats     */
td[class~="n"]  { color: #AAAAAA; }              /* Tag des Folgemonats   */

/* Abschnitt mit Verweisen am oberen Rand */
div#bt a {
    cursor: pointer;
    color: inherit;
    text-decoration: none;
}
div#bt a + a {
    margin-left: 3em;
}

/* Schaltfläche "nach links" */
div#bl {
    text-align: left;
    bottom: 0;
    cursor: pointer;
}

/* Schaltfläche "nach rechts" */
div#br {
    text-align: right;
    bottom: 0;
    cursor: pointer;
}
```

```
/* Randfelder, die nur bei Bedarf sichtbar sind */
div[class~="n"] {
    display: block;
    position: fixed;
    top: 0;
    color: transparent;
    background-color: transparent;
    border: none;
    margin: 0;
    padding: 0;
    font-size: 200% ;
}


/* Zeiger über einem Randfeld */
div[class~="n"]:hover {
    color: #000000;
}


/* Fenster "Einstellung" */
div#set {
    display:  block;
    position: fixed;
    top: 10px;
    left: 10px;
    background-color: #d9cd84;
    color: #000000;
    border-style: solid;
    border-width: 1px;
    padding: 1em;
}


div#set[class~="hide"] {
    display: none;
}


p[class~="lft"] { text-align: left; }
input { margin-right: 1em; }


/*
  möglich für Übersicht Vormonat, Übersicht Folgemonat,
  Spalte "Monat", Spalte "Kalenderwoche"
*/
*[class~="hide"] {
    display: none;
}


a {
  outline: none;
  padding: 0 10px;
}
img { border: none; }
```

# kal.js

[Quelltext]

Dieses Skript erlaubt der Surferin,

durch die „Kalenderblätter” zu blättern,
und die Darstellung ein wenig anzupassen, wie in der Hilfe beschrieben.

Das „Blättern” erfolgt, indem die Eigenschaften `display` und `visibility` der betroffenen Elemente gesetzt werden. Da hier zwei verschiedene Sätze von CSS-Regeln (für die Bildschirmansicht und den Druck) benutzt werden, muss ich besonders vorsichtig sein. Die direkten Eigenschaften beeinflussen beide Darstellungen. Im Hinblick auf mehrere CSS-Regelsätze (z.B. für Bildschirme, Mobilgeräte und Drucker) ist es sinnvoll, CSS-Eigenschaften nicht direkt per Javascript zu setzen, sondern geeignete Attributwerte zu setzen.

Ich habe mich dennoch entschieden, CSS-Eigenschaften direkt zu setzen. Das angezeigte Kaldenderblatt (`div`-Element) hat die Eigenschaft `display: block`. Wenn die Surferin „blättert”, darf ich nicht `display: none` setzen, weil dann das Blatt nicht gedruckt würde, sondern muss die `display`-Eigenschaft entfernen. Auf dem Deckblatt ist die Schaltfläche „zurück” (links) verborgen, auf dem Dezember-Blatt ist die Schaltfläche „weiter” (rechts) verborgen. Um die Schaltflächen anzuzeigen, darf ich nicht deren `display`-Eigenschaft zum Beispiel auf `block` setzen, weil sie dann auch im Ausdruck erschienen.

# Quelltext

[Beschreibung]

```
// -*- coding: utf-8 -*-
/*
  Script zur Kalender-Ansicht
  2015-12-14 Herbert Schiemann <h.schiemann@herbaer.de>
*/
onload = function () {

    // Anpassung der Darstellung
    var rh = new RegExp ("\\s*\\bhide\\b");

    // Hilfsvariable
    var i;   // lfd. Zähler
    var id;  // ID eines Elements
    var l;   // Knotenliste oder andere Liste
    var n;   // ein Element / Knoten
    var v;   // Element-Zustand / Eingabewert / Wertepaar
    var c;   // Wert des Attributs @class
    var r;   // regulärer Ausdruck zu einem "Wort" in @class

    // gespeicherte Einstellungen laden
    var sto = window.localStorage || window.sessionStorage;
    if (sto) {
        l = sto.getItem ("kal");
        if (l) {
            l = String (l).split (":");
            for (i = 0; i < l.length; ++i) {
                v = l[i].split("=");
                if (v.length == 2) {
                    id = v[0];
                    document.getElementById (id).checked = (v[1] == "1");
/*
                    n = document.getElementById (id);
                    if (n) {
                        if (id.substring(0, 4) == "chk_")
                            n.checked = (v[1] == "1");
                        else
                            n.value = v[1];
                    }
*/
                }
            }
        }
    }

    // Fenster "Einstellung" anzeigen
    var set = function (e) {
        e.stopPropagation ();
        n = document.getElementById ("set");
        c = n.getAttribute ("class");
        n.setAttribute ("class", c.search (rh) >= 0 ? c.replace (rh, "") : c + " hide");
    }; // set
    document.getElementById ("as").addEventListener ("click", set, false);

    // IDs der Checkboxen, deren Einstellungen zu speichern sind
    var ids = ["chk_vm", "chk_fm", "chk_kw", "chk_mc"];

    // "Einstellung" schließen und Einstellungen speichern
    var save = function (e) {
        e.stopPropagation ();
        n = document.getElementById ("set");
        c = n.getAttribute ("class");
        n.setAttribute ("class", c + " hide");
        if (sto) {
            v = "";
            for (i = 0; i < ids.length; ++i) {
                id = ids[i];
                v += (i ? ":" : "") + id + "="
                    + (document.getElementById (id).checked ? "1" : "0");
/*
                n = document.getElementById (id);
                v += (i ? ":" : "") + id + "=" +
                    (id.substring(0, 4) == "chk_" ?
                     (n.checked ? "1" : "0") : n.value);
*/
            }
            sto.setItem ("kal", v);
        }
    };
    document.getElementById ("b_cl").addEventListener ("click", save, false);

    // Checkbox
    var chk = function (id, tag, cw) {
        var b = document.getElementById ("chk_" + id);  // Eingabeelement
        var r = new RegExp ("\\b" + cw + "\\b");
        // event handler
        var h = function (e) {
            if (e) e.stopPropagation ();
            v = b.checked ? 1 : 0;
            l = document.getElementsByTagName (tag);
```

```
        for (i = 0; i < l.length; ++i) {
            n = l[i];
            c = n.getAttribute ("class");
            if (c && c.search (r) >= 0)
                n.setAttribute ("class", v ? c.replace (rh, "") : c + " hide");
        }
    }
    h ();
    b.addEventListener ("change", h, false);
}; // chk

chk ("vm", "table", "v");
chk ("fm", "table", "n");
chk ("kw", "td",    "kw");
chk ("mc", "td",    "mc");

// Blättern

// der aktuelle Abschnitt
var cd = document.getElementById ("d00");
// die Nummer des aktuellen Abschnitts: 0 Deckblatt, 1 Januar, 12 Dezember
var cn = 0;
// linke Seite: Verweis auf das vorhergehenden "Blatt"
var bl = document.getElementById ("bl");
// rechte Seite: Verweis auf das folgende "Blatt"
var br = document.getElementById ("br");
// Verweis zum Deckblatt
var ah = document.getElementById ("ah");

cd.style.display    = "block";
ah.style.visibility = "hidden";

// Elemente positionieren
n = document.getElementsByTagName ("body")[0];
v = parseFloat (window.getComputedStyle (n, "").getPropertyValue ("width"));
c = Math.floor (v / 10);
v = Math.floor (v);
bl.style.width   = c + "px";
bl.style.display = "none";
br.style.left    = (v - c) + "px";
br.style.width   = c + "px";
document.getElementById ("bt").style.width = v + "px";

// zum vorhergehenden Blatt
var on_bl = function (e) {
    e.stopPropagation ();
    if (cn > 0) {
        --cn;
        id = "a" + (100 + cn);
        id = "d" + id.slice (2);
        cd.style.display = "";
        cd = document.getElementById (id);
        cd.style.display = "block";
        if (cn == 0) {
            bl.style.display = "none";
            ah.style.visibility = "hidden";
        }
        else if (cn == 11) {
            br.style.display = "";
        }
    }
}; // on_bl
bl.addEventListener ("click", on_bl, false);

// zum nächsten Blatt
var on_br = function (e) {
    e.stopPropagation ();
    if (cn < 12) {
        ++cn;
        id = "a" + (100 + cn);
        id = "d" + id.slice (2);
        cd.style.display = "";
        cd = document.getElementById (id);
        cd.style.display = "block";
        if (cn == 12) {
            br.style.display = "none";
        }
        else if (cn == 1) {
            bl.style.display = "";
            ah.style.visibility = "";
        }
    }
}; // on_bl
br.addEventListener ("click", on_br, false);

// von einem Bild auf dem Deckblatt zum Monatsblatt
var on_sel = function (e) {
    e.stopPropagation();
    c = e.target.getAttribute("id").slice(1);
    n = document.getElementById("d" + c);
    n.style.display = "block";
    cd.style.display = "";
    cd = n;
    if (cn == 0)
        ah.style.visibility = "visible";
    cn = Number (c);
```

```
        bl.style.display = ( cn == 0  ? "none" : "" );
        br.style.display = ( cn == 12 ? "none" : "" );
    }; // on_sel
    l = cd.getElementsByTagName ("img");
    for (i = 0; i < l.length; ++i) {
        l[i].addEventListener ("click", on_sel, true);
    }

    // zum Deckblatt
    var on_h = function (e) {
        e.stopPropagation();
        if (cn) {
            n = document.getElementById("d00");
            n.style.display = "block";
            cd.style.display = "";
            cd = n;
            cn = 0;
            bl.style.display = "none";
            br.style.display = "";
            ah.style.visibility = "hidden";
        }
    }; // on_h
    document.getElementById ("ah").addEventListener ("click", on_h, false);
} // onload
```

# prt.css

[Quelltext]

## Allgemein

Diese Datei wird von der Transformation `kal.xslt` eingebunden. Sie ist für eine Ansicht vorgesehen, die die Tage einer Woche in einer waagerechten Zeile anzeigt. Sie entspricht weitgehend den Regeln zur Bildschirmansicht (`kal.css`), nur sind die „interaktiven” Elemente verborgen.

Die Breiten sind so festgelegt, dass die Druckausgabe mit einem ziemlich alten Browser funktioniert. Für Firefox 60 (und wahrscheinlich viele andere Browser) sind die Werte in den Kommentaren geeignet.

## Textfarbe

Mein alter Browser stellt in der Druckausgabe zwar Feiertage richtig in Rot dar, aber keine Hintergrundfarben und Graustufen für den Text. Daher habe ich für Tage des Vormonats und des Folgemonats eine „transparente” Farbe vorgesehen.

# Quelltext

[Beschreibung]

```
/*
  Stil für Kalender, Druck
  2018-11-03 Herbert Schiemann <h.schiemann@herbaer.de>
  2020-12-31 angepasst an Bilder im Format 2:3 mit aktueller Firefox-Version
  2022-02-13 body font-size angepasst
*/


body {
    position: absolute;
    width: 100%;
    /* muss/kann ggf. angepasst werden */
    font-size: 14.7pt;
}


div { text-align: center; }


/* Die "Kalenderblätter" erscheinen auf einzelnen Seiten */
body > div + div {
    page-break-before: always;
    page-break-inside: avoid;
}


img[class~="d"] {
    width: 20%;
    margin: 2% 2%;
}


h1 { font-family: sans-serif; }


img[class~="m"] {
  /*
    width: 65%;
    margin: 1em;
 */
  /* für neuere Firefox-Version: */
  width: 70%;
  margin-bottom: 1em;
}


table {
    width: 95%;
    margin: 0 2%;
}


table[class~="v"] { background-color: #EEDDDD; } /* Vormonat   */
table[class~="n"] { background-color: #DDEEDD; } /* Folgemonat */


*[class~="hide"] { display: none; }             /* verborgene Elemente */


td[class~="kw"] { font-size: 80%;   width: 2em;} /* Kalenderwoche */
td[class~="mc"] { text-align: left; width: 6em;} /* Monatsname    */
td[class~="f"]  { color: #FF0000; }              /* Feiertag      */
td[class~="w7"] { color: #FF0000; }              /* Sonntag       */
td[class~="p"]  { color: transparent; }          /* Tag des Vormonats   */
td[class~="n"]  { color: transparent; }          /* Tag des Folgemonats */


div[class~="n"] { display: none; }              /* Interaktive Elemente  */
div#set         { display: none; }              /* Fenster "Einstellung" */


a   { outline: none; }
img { border: none;  }
```

# Datei kal_help.xhtml.de

```
<?xml version="1.0" encoding="utf-8"?>
<!-- file KLEIDER/web/src/kalender/kal_help.xhtml.de -->
<!--
  Hilfe zum Kalender
  2015 Herbert Schiemann <h.schiemann@herbaer.de>
  Borkener Str. 167, 46284 Dorsten, Germany
-->
<html
   xmlns    = "http://www.w3.org/1999/xhtml"
   xmlns:l  = "http://herbaer.de/xmlns/20141210/localization"
   xml:lang = "de"
>
<head>
  <meta name="viewport" content="width=device-width, initial-scale=1"/>
  <link href = "/style/shortcut_icon.png" rel = "icon"/>
  <link href = "kal_help.css" rel = "stylesheet"/>
  <script src = "kal_help.js"/>
  <title>Der Kalender</title>
</head>
<body>
  <h1>Der Kalender</h1>
  <div>
    <!-- Allgemeines -->
    <p>
Diese Website <a href = "/">kleider.herbaer.de</a> wird Ihnen angeboten von
"Herbär" Herbert Schiemann
(<a href="http://herbaer.de">http://herbaer.de</a>,
<a href="mailto:h.schiemann@herbaer.de">mailto:h.schiemann@herbaer.de</a>).
    </p>
    <p>
Ich kann nicht alle Feiertage in allen Ländern der Welt kennzeichnen,
sondern nur die Feiertage für einige wenige Länder.
Ich kann nicht garantieren, dass alle Feiertage richtig angezeigt werden.
Die angezeigten Kalenderdaten hängen ab von der Spracheinstellung
mit Länderkennzeichnung des Browsers.
Wenn keine passenden Kalenderdaten auf meiner Website verfügbar sind,
werden die Kalenderdaten gemäß der Spracheinstellung "de-de" angezeigt:
die Monatsnamen erscheinen in deutscher Sprache, und die bundesweiten
Feiertage in Deutschland sind gekennzeichnet.
    </p>
    <p>
Sie können den Kalender (hoffentlich) einfach ausdrucken.
Ggf. müssen Sie die Skalierung oder die Seiteneinrichtung anpassen,
damit immer ein Monat auf einer Seite ausgedruckt wird.
    </p>
  </div>
  <div>
    <!-- Hilfe zur Navigation -->
    <p>
In der Ansicht im Browser wird zunächst das "Deckblatt" angezeigt.
Ein Klick auf ein Bild im Deckblatt zeigt die Kalenderseite zu dem entsprechenden Monat an.
Ein Klick auf das Bild einer Monatsseite führt zu der Bildergeschichte,
der das Bild entnommen ist.
    </p>
    <p>
Wenn der Mauszeiger an den oberen, den linken oder den rechten Rand der Anzeige kommt,
werden die folgenden Symbole sichtbar:
    </p>
    <table>
      <tbody>
        <tr>
          <td class = "s">&#x25c9;</td>
          <td>
Dieses Zeichen erscheint oben auf einer Monatsseite.
Ein Klick auf das Zeichen zeigt das Deckblatt an.
          </td>
        </tr>
        <tr>
          <td class = "s">&#x2302;</td>
          <td>
Das Zeichen am oberen Rand verweist auf die Startseite der Website.
          </td>
        </tr>
        <tr>
          <td class = "s">&#x22ee;</td>
          <td>
Ein Klick auf dieses Zeichen am oberen Rand zeigt das Fenster
&#x201e;<span class = "oll" id = "oll_set"><l:ph id = "einstell"/></span>&#x201d; an.
Ein zweiter Klick auf dieses Zeichen schließt das Fenster
&#x201e;<span class = "oll" id = "oll_set"><l:ph id = "einstell"/></span>&#x201d;,
ohne die geänderten Einstellungen zu speichern.
          </td>
        </tr>
        <tr>
          <td class = "s">?</td>
          <td>
Das Fragezeichen am oberen Rand verweist auf diese Hilfe.
          </td>
        </tr>
        <tr>
          <td class = "s">&#x25c0;</td>
```

```
          <td>
Wenn dieses Zeichen links oben zu sehen ist,
zeigt ein Mausklick das Kalenderblatt des Vormonats an.
Fall das Kalenderblatt des Januar angezeigt wird,
zeigt ein Mausklick das Deckblatt an.
          </td>
        </tr>
        <tr>
          <td class = "s">&#x25b6;</td>
          <td>
Wenn dieses Zeichen rechts oben zu sehen ist,
zeigt ein Mausklick das Kalenderblatt des Folgemonats an.
Vom Deckblatt führt ein Mausklick zum Kalenderblatt des Monats Januar.
          </td>
        </tr>
      </tbody>
    </table>
    <p>
Die Tage einer Woche werden in einer waagerechten Reihe angezeigt.
Der erste Tag ist ein Montag.
Sonntage und Feiertage sind durch rote Schrift gekennzeichnet.
Tage des Vormonats oder des Folgemonats erscheinen in grauer Schrift.
    </p>
  </div>
  <div>
    <!-- Hilfe zu Einstellungen -->
    <div id="set">
      <h3><l:ph id = "einstell"/></h3>
      <p class = "lft">
        <label>
          <input type = "checkbox" id = "chk_vm"/>
          <l:ph id = "vormonat"/>
        </label>
      </p>
      <p class = "lft">
        <label>
          <input type = "checkbox" id = "chk_fm"/>
          <l:ph id = "folgemonat"/>
        </label>
      </p>
      <p class = "lft">
        <label>
          <input type = "checkbox" id = "chk_kw" checked = "checked"/>
          <l:ph id = "kalenderwoche"/>
        </label>
      </p>
      <p class = "lft">
        <label>
          <input type = "checkbox" id = "chk_mc"/>
          <l:ph id = "monatsname_in_tabelle"/>
        </label>
      </p>
      <p>
        <button id = "b_cl"><l:ph id = "schliessen"/></button>
      </p>
    </div>
    <p>
Wenn Sie wünschen, werden auf einem Kalenderblatt auch die Kalenderwochen
und die Tage des Vormonats oder des Folgemonats angezeigt.
Beim Ausdrucken müssen Sie auf die Seiteneinrichtung achten (Ränder, Skalierung,
Seitengröße).
    </p>
    <p>
Ein Klick auf das Zeichen &#x22ee; am oberen Rand zeigt das Fenster
&#x201e;<span class = "oll" id = "oll_set"><l:ph id = "einstell"/></span>&#x201d; an.
    </p>
    <p>
Die Markierung &#x201e;<span class = "oll" id = "oll_chk_vm"><l:ph id = "vormonat"/></span>&#x201d;
zeigt über dem aktuellen Monat die Übersicht des Vormonats an.
    </p>
    <p>
Die Markierung &#x201e;<span class = "oll" id = "oll_chk_fm"><l:ph id = "folgemonat"/></span>&#x201d;
zeigt unter dem aktuellen Monat die Übersicht des Folgemonats an.
    </p>
    <p>
Die Markierung &#x201e;<span class = "oll" id = "oll_chk_kw"><l:ph id = "kalenderwoche"/></span>&#x201d;
zeigt links neben jeder Woche die Nummer der Woche im Kalenderjahr an.
    </p>
    <p>
Die Markierung &#x201e;<span class = "oll" id = "oll_chk_mc"><l:ph id = "monatsname_in_tabelle"/></span>&#x201d;
zeigt links neben der Übersicht eines Monats den Namen des Monats an.
    </p>
    <p>
Ein Klick auf die Schaltfläche
&#x201e;<span class = "oll" id = "oll_b_cl">Schließen</span>&#x201d;
schließt das Fenster und speichert die Markierungen (lokal,
Ihre Einstellungen verlassen nicht den Rechner).
Wenn die Einstellungen Ihres Browsers es zulassen, bleiben die Markierungen
erhalten, wenn Sie wieder eine Kalender dieser Website öffnen.
    </p>
  </div>
</body>
</html>
```

# kal_help.css

[Quelltext]

# Die Aufgabe dieser Datei

Diese Datei (`kal_help.css`) wird von der Hilfedatei `kal_help.xhtml.de` eingebunden. Sie enthält Regeln zur Darstellung des Fensters „Einstellung” entsprechend den Regeln in `kal.css` für die Kalender.

Sie enthält außerdem Regeln zur Hervorhebung der beschriebenen Elemente und Regeln für die Hinweise zur maschinellen Übersetzung.

# Quelltext

[Beschreibung]

```
/*
  Darstellung der Hilfe zum Kalender
  2015-12-14 Herbert Schiemann <h.schiemann@herbaer.de>
*/


/* Tabelle zu den Zeichen am Rand */
td              { vertical-align: middle; }
td[class~="s"]  { font-size: 200%; padding: 0 10px;}


/* Fenster "Einstellung" */
div#set {
    margin: 1em;
    float: left;
    background-color: #d9cd84;
    color: #000000;
    border-style: solid;
    border-width: 1px;
    padding: 1em;
    text-align: center;
}
p[class~="lft"] { text-align: left; }
input { margin-right: 1em; }


/* Verweise auf Elemente der Darstellung */
span[class~="oll"]       { text-decoration: underline; }
span[class~="oll"]:hover { background-color: #FFCCCC; color: #000000; }


/* Die aktuell hervorgehobenen Elemente */
*[class~="outline"] {
    outline: #FF8888 solid 6px;
    background-color: #FFDDDD;
    color: #000000;
}


/* Kein Rahmen um Bilder, die als Verweise dienen */
a img { border: none; }


/* Google-Bild und Texthinweis nebeneinander */
div[class="machine"] p {
    display: inline-block;
    vertical-align: middle;
    margin-right: 1em;
}
```

# kal_help.js

[Quelltext]

Dieses Skript hebt das bezeichnete Element hervor, wenn der Mauszeiger in der Hilfe auf die Bezeichnung eines (Eingabe-)Elements zeigt.

# Quelltext

[Beschreibung]

```
/*
  Hervorhebungen in der Hilfe zum Kalender
  2016-01-04 Herbert Schiemann <h.schiemann@herbaer.de>
*/
onload = function () {
   var id;                   // ID des referenzierten Elements
   var r  = /\s+outline\b/ ;
   var f  = /^.*?_(.*)/    ;  // ID-"Filter"
   var e ;                   // referenziertes Element
   var c ;                   // Wert des Attributs class

   var over = function (ev) {
      id  = ev.target.getAttribute("id").match (/^.*?_(.*)/)[1];
      e = document.getElementById (id);
      e.setAttribute ("class", e.getAttribute ("class") + " outline");
   }; // over

   var out = function (ev) {
      id = ev.target.getAttribute("id").match (/^.*?_(.*)/)[1];
      e  = document.getElementById (id);
      c  = e.getAttribute ("class");
      e.setAttribute ("class", c.replace (r, ""));
   }; // out

    var i;
    var s = document.getElementsByTagName ("span");
    for (i = 0; i < s.length; ++i) {
        if (s[i].getAttribute ("class") == "oll") {
            s[i].addEventListener ("mouseover", over, false);
            s[i].addEventListener ("mouseout",  out,  false);
        }
    }
};
```

# Einbindung in die Website

Wie ist die Kalenderansicht einer Bildergeschichte in die Website eingebunden? Wie sind die Kalender mit „von Hand” zusammengestellten Bildern in die Website eingebunden? Sind weitere Teile der Website betroffen?

Die Kalenderansicht einer Bildergeschichte liegt als weitere Ansicht unter dem URI `/s`*`STORY`*`/kal`.

Die URI der Kalender sind die URI der Bildauswahldateien: `/kal/`*`JAHR`*`/`*`BILDAUSWAHL`*`.xml`. Unter der URI `/kal/`*`JAHR`*`/index.xhtml` liegt eine Index-Datei mit Verweisen auf die Kalender des Jahres. Unter der URI `/kal/index.xhtml` liegt eine Index-Datei mit Verweisen auf alle Jahre und alle Kalender.

Der Menübaum der Startseite enthält Verweise auf die Kalender und die Kalender-Index-Dateien. Die Einträge im Menübaum werden aus der Datei `kalender/tree.xml` (s. unten) gelesen. Der Aufbau der Startseite ist im Zusammenhang mit dem allgemeinen Stil der Website beschrieben.

Aus dem Inhalt des Verzeichnisses `kalender` erstellen das Skript `tree.pl` und die anschließende Transformation `tree_sorttlv.xslt` die Datei `kalender/tree.xml`. Sie enthält die Baumstruktur für den Teilbaum der Startseite, XML-Namensraum `http://herbaer.de/xmlns/20151222/tree/ [tree.rng]`.

Wenn Dateien `kalender/`*`JAHR`*`/index.xhtml.`*`LANG`* oder `kalender/index.xhtml.`*`LANG`* existieren, werden die Index-Dateien aus diesen erzeugt. Die Transformation `xhtml_setlinks.xslt` fügt Verweise auf das Website-Icon und CSS-Regeln für eingebetteten Inhalt ein. Die Transformation `xhtml_target.xslt` bewirkt, das die Ziele der Verweise (die Kalender) in einem neuen Fenster geöffnet werden.

Falls eine Datei `kalender/`*`JAHR`*`/index.xhtml.de` oder `kalender/index.xhtml.de` nicht existiert, erzeugt `tree_ht.xslt` die deutschsprachigen Index-Dateien aus `tree_ht.xslt`. Die so erzeugten Index-Dateien nutzen zur „Baumansicht” der verschachtelten Listen die Transformation `treelist.xslt` in Verbindung mit `treelist.js` und `treelist.css`.

Die Datei `kalender/tree.xml` bestimmt, welche Index-Dateien und welche Bildauswahl-Dateien im Server-Verzeichnis angelegt werden. Die Transformation `tree_cmd.xslt` erzeugt die Befehle zum Einfügen der Index-Dateien, die Transformation `tree_files.xslt` liefert eine Liste der relativen URI der einzufügenden Bildauswahldateien, s. Bash-Skript `kalender`.

Zu Einbindung in die Website im weiteren Sinne gehören auch Schlüsselwörter für Suchmaschinen.

# tree.pl

[Quelltext]

# Übersicht

`tree.pl --help|--version`

`tree.pl [ --verbose ... | --no_verbose ]`
`[ --lang` *`LANG`* `][ --webdir` *`WEBDIR`* `][ --kaldir` *`KALDIR`* `][ --srcdir` *`SRCDIR`* `]`
`[ --shiftnd` *`SHIFTND`* `]...`
`[ --incldir` *`INCLDIR`* `]... [ --excldir` *`EXCLDIR`* `]...`
`[ --inclfile` *`INCLFILE`* `]... [ --exclfile` *`EXCLFILE`* `]...`
`[ --rtname` *`RTNAME`* `][ --rtrefbase` *`RTREFBASE`* `][ --rttitle` *`RTTITLE`* `][ --ixfile` *`IXFILE`* `]`
`[ --dirnmflt` *`DIRNMFLT`* `]... [ --filenmflt` *`FILENMFLT`* `]...`
`[ --titlecmd` *`TITLECMD`* `][ --xmlns` *`XMLNS`* `]`

# Beschreibung

Dieses Programm erzeugt die Datei `kalender/tree.xml` (s. `tree.rng`) mit der Baustruktur der Kalender, wie sie in die Startseite der Website eingebunden wird. Zum Verzeichnis `kalender` wird der „Wurzelknoten" (der Kalender) angelegt. Die Kalender eines Jahres *`JAHR`* liegen im Unterverzeichnis `kalender/`*`JAHR`*. Zu jedem Jahr wird ein Verzeichnisknoten mit den Kalendern als Dateiknoten (Endknoten) angelegt, zusätzlich werden zu den Kalendern des „aktuellen" (s. `--shiftnd`) Jahres Dateiknoten direkt unter dem Wurzelknoten angelegt. In der Voreinstellung ist das aktuelle Jahr in den Monaten Januar bis November das laufenden Jahr, im Dezember das Folgejahr.

Es ist mit Perl 5.10.1 getestet.

# Optionen

Zu allen Befehlszeilenargumenten gibt es Voreinstellungen.

Zum Beispiel wählt „`--incldir .`" zunächst alle Unterverzeichnisse von *`KALDIR`* aus. „`--excldir` '/no$'" schließt nur Unterverzeichnisse mit dem Namen `no` aus. Wenn es kein Unterverzeichnis mit dem Namen `no` gibt, werden alle Unterverzeichnisse und deren Unterverzeichnisse ausgewählt.

Entsprechend wählt „`--inclfile .` `--exclfile` /whisky\$" alle Dateien aus, die nicht `whisky` heißen.

`--help`

Gibt eine kurze Hilfe mit den Voreinstellungen zu allen möglichen Befehlszeilenargumenten aus.

`--version`

Gibt kurze Hinweise zum Programm und die Version aus.

`--verbose`

Erhöht den Umfang der Meldungen nach STDERR.

`--no_verbose`

Unterdrückt die Ausgabe von Meldungen. Die Optionen `--verbose` und `--no_verbose` werden der Reihe nach ausgewertet.

`--lang` *`LANG`*

*`LANG`* ist die Kennung der Sprache mit möglichen Erweiterungen. Der Wert wird für den Platzhalter *`${lang}`* in den Befehl *`TITLECMD`* zur Ermittlung des Titels eines Dokuments eingesetzt.

`--webdir` *`WEBDIR`*

Das Basis-Verzeichnis zur Website wird für den Platzhalter *`${webdir}`* in den Werten *`KALDIR`*, *`SRCDIR`* und *`TITLECMD`* eingesetzt.

`--kaldir` *`KALDIR`*

Das Verzeichnis der Kalender-Quelldaten mit dem Unterverzeichnis `base` für die Grunddaten und Unterverzeichnissen *`JAHR`* für die Bildauswahl-Dateien der Kalender eines Jahres. Der Wert kann Platzhalter *`${xxx}`* (z.B. *`${webdir}`*) für das Befehlszeilenargument *`XXX`* enthalten.

`--srcdir` *`SRCDIR`*

Das Verzeichnis der Skripte und Quellcode-Dateien zun dem Kalendern. Der Wert kann Platzhalter *`${xxx}`* (z.B. *`${webdir}`*) für das Befehlszeilenargument *`XXX`* enthalten.

`--shiftnd` *`SHIFTND`*

Der Name eines Knotens (s. `name`), dessen Kindknoten zu Kindknoten des Wurzelknotens kopiert werden sollen. Das gesamte Argument kann wiederholt werden. Dann werden möglicherweise die Unterknoten mehrerer Knoten kopiert.

`--incldir` *`INCLDIR`*

Die Unterverzeichnisse von *`KALDIR`*, die in die ausgegebene Baumstruktur eingehen, werden in zwei Schritten ausgewählt. Im ersten Schritt werden alle Unterverzeichnisse ausgewählt, deren vollständiger Verzeichnispfad einem der regulären Ausrücke *`INCLDIR`* entspricht. Das gsamte Argument kann wiederholt werden. Im zweiten Schritt werden alle Unterverzeichnisse ausgenommen, deren Verzeichnispfad einem der regulären Ausdrücke *`EXCLDIR`* entspricht.

`--excldir` *`EXCLDIR`*

Ein Unterverzeichnis von *`KALDIR`*, dessen vollständiger Verzeichnispfad einem der regulären Ausdrücke *`EXCLDIR`* entspricht (das gesamte Argument kann wiederholt werden), wird nicht in die ausgegebene Baumstruktur aufgenommen, auch wenn der Verzeichnispfad einem der Ausrücke *`INCLDIR`* entspricht.

`--inclfile` *`INCLFILE`*

Dateien im Verzeichnis *`KALDIR`* oder einem ausgewählten Unterverzeichnis (s. *`INCLDIR`* und *`EXCLDIR`*) werden vorläufig ausgewählt, wenn ihr vollständiger Dateipfad einem regulären Ausdruck *`INCLFILE`* entspricht. Das Argument kann mehrfach vorkommen. Wenn der Dateipfad dann keinem der regulären Ausdrücke *`EXCLFILE`* entspricht, wird die Datei in die Baumstruktur aufgenommen.

`--exclfile` *`EXCLFILE`*

Der reguläre Ausdruck *`EXCLFILE`* beschreibt Dateipfade, die aus der ausgegebenen Baumstruktur ausgeschlossen sind (s. *`INCLFILE`*). Das gesamte Argument kann mehrfach vorkommen.

`--rtname` *`RTNAME`*

*`RTNAME`* ist der Name (s. `name`), des Wurzelknotens, der dem Verzeichnis *`KALDIR`* entspricht.

`--rtrefbase` *`RTREFBASE`*

Der Wert *`RTREFBASE`* wird als Verweis-Basis (s. `refbase`) zum Wurzelknoten ausgegeben.

`--rttitle` *RTTITLE*

Der Titel (s. `title`) des Wurzelknotens, der dem Verzeichnis *KALDIR* entspricht.

Wenn die Datei *KALDIR/IXFILE* existiert, wird der voreingestellte Titel dieser Datei entnommen (s. *TITLECMD*).

`--ixfile` *IXFILE*

Der Name der Index-Datei eines Verzeichnisses. Der Wert *IXFILE* hat zwei Funktionen: er wird als Inhalt des Elements `ref` zu Verzeichnis-Knoten ausgegeben. Wenn die Datei mit dem Namen *IXFILE* im Verzeichnis *KALDIR* oder einem ausgewählten Unterverzeichnis existiert, dann wird aus dieser Datei der Titel des Verzeichnisknotens bestimmt (s. *TITLECMD*).

`--dirnmflt` *DIRNMFLT*

Der Name (`name`) eines Verzeichnisknotens wird aus dem Verzeichnispfad bestimmt. Er ist die erste "Matchgruppe" des regulären Ausdrucks *DIRNMFLT*, der auf den vollständigen Verzeichnispfad angewandt wird. Das gesamte Argument kann mehrfach vorkommen. Der erste passende reguläre Ausdruck bestimmt den Namen. Wenn kein regulärer Ausdruck passt, ist der Verzeichnisname auch der Knotenname.

`--filenmflt` *FILENMFLT*

Der Name (`name`) eines Dateiknotens wird aus dem Dateipfad bestimmt. Er ist die erste "Matchgruppe" des regulären Ausdrucks *DIRNMFLT*, der auf den vollständigen Dateipfad angewandt wird. Das gesamte Argument kann mehrfach vorkommen. Der erste passende reguläre Ausdruck bestimmt den Namen. Wenn kein regulärer Ausdruck passt, ist der Dateiname auch der Knotenname.

`--titlecmd` *TITLECMD*

Befehl zur Bestimmung des Titels eines Dokuments mit Platzhaltern. Die möglichen Platzhalter sind `${lang}`, `${webdir}`, `${kaldir}`, `${srcdir}` und `${ixfile}` für die Befehlszeilenargumente sowie `${file}` für den Dateipfad des Dokuments, dessen Titel bestimmt werden soll.

Voreingestellt ist das Bash-Skript `titlecmd`.

`--xmlns` *XMLNS*

*XMLNS* ist der XML-Namensraum der Ausgabe.

# Benutzte Module

Das Programm benutzt die folgenden Module:

`Herbaer::Readargs` (`Readargs.pm`)

Die Funktion `Herbaer::Readargs::read_args` liest die Befehlszeilen-Argumente.

`Herbaer::Replace`

Die Funktion `Herbaer::Replace::replace` ersetzt Platzhalter der Form `${xxx}`. Die Datei `Replace.pm` ist im Zusammenhang mit Lokalisierungen / Übersetzungen beschrieben.

`Herbaer::XMLDataWriter` (Datei `XMLDataWriter.pm`)

Ausgabe von XML-Daten.

# Quelltext

[Beschreibung]

```
#!/usr/bin/perl -w
# Baumstruktur der Kalender
# 2015-12-22 Herbert Schiemann <h.schiemann@herbaer.de>
# 2020-04-12 Voreinstellung webdir


use utf8 ;              # Dieser Quelltext ist utf-8-kodiert
use Cwd qw(realpath);
use File::Spec::Functions;
use Herbaer::Readargs;   # read_args ()
use Herbaer::Replace;    # replace ()
use Herbaer::XMLDataWriter ;
# use POSIX;


binmode (STDERR, ":encoding(utf-8)");
binmode (STDOUT, ":encoding(utf-8)");


my $args = {
   "[cnt]verbose" => 1,
   "lang"         => "de",                       # Sprache
   "webdir"       => undef,                      # Website-Basisverzeichnis
   "kaldir"       => '${webdir}/kalender',       # Verzeichnis der Kalender-Quellen
   "srcdir"       => '${webdir}/src/kalender',   # Verzeichnis der Skripte
   "shiftnd"      => [],                         # Namen "geschobener"  Knoten
   "incldir"      => [],                         # Regex eingeschlosser Verzeichnisse
   "excldir"      => [],                         # Regex ausgenommener Verzeichnisse
   "inclfile"     => [],                         # Regex eingeschlossener Dateien
   "exclfile"     => [],                         # Regex ausgenommener Dateien
   "rtname"       => undef,                      # Name des Wurzelknotens
   "rtrefbase"    => "kal",                      # Verweis-Basis des Wurzelknotens
   "rttitle"      => undef,                      # Titel des Wurzelknotens
   "ixfile"       => "index.xhtml",              # Default-Index-Datei
   "dirnmflt"     => [],                         # Namensfilter für Verzeichnisse
   "filenmflt"    => [],                         # Namensfilter für Dateien

   # Befehl zur Bestimmung des Titels mit Platzhaltern
   "titlecmd"     => "\${srcdir}/titlecmd \${file} \${lang}",

   # XML-Namensraum der Ausgabe
   "xmlns"        => "http://herbaer.de/xmlns/20151222/tree/",
};


my $data = {};


sub set_default {
   my $args = shift;
   if (! $args -> {"webdir"}) {
      my $b = realpath ($0);
      $b =~ s/\/src\/kalender\/tree\.pl//;
      $args -> {"webdir"} = $b;
   }
   my $k;
   for $k ("kaldir", "srcdir", "titlecmd") {
      $args -> {$k} = replace ($args -> {$k}, $args);
   }
   if (! @{$args -> {"shiftnd"}} ) {
      my $lt = [localtime()];
      my $m = $lt -> [4];
      my $y = $lt -> [5] + 1900;
      ++$y if $m == 11;
      $args -> {"shiftnd"} = [ "$y", ];
   }
   @{$args -> {"incldir"}}   or $args -> {"incldir"}   = ['/2\d{3}$',];
   @{$args -> {"excldir"}}   or $args -> {"excldir"}   = ['/feiertage$',];
   @{$args -> {"inclfile"}}  or $args -> {"inclfile"}  = ['\.xml\.?$',];
   @{$args -> {"exclfile"}}  or $args -> {"exclfile"}  = ['/tree\.xml$',];
   @{$args -> {"dirnmflt"}}  or $args -> {"dirnmflt"}  = ['^(?:.*/)*([^.]+)',];
   @{$args -> {"filenmflt"}} or $args -> {"filenmflt"} = ['^(?:.*/)*([^.]+)',];

   my $re;  # regular expression
   my $rel; # regular expression list
   if (! $args -> {"rtname"}) {
      $args -> {"rtname"} = "kalender";
      $rel = [ map { qr/$_/ } @{$args -> {"dirnmflt"}} ];
      for $re (@$rel) {
         if ( $args -> {"kaldir"} =~ $re ) { $args -> {"rtname"} = $1; last; }
      }
   }
   my $pth;   # Dateipfad
   my $cd;    # Befehl zum Ermitteln des Titels
   my $ch;    # Handle zum Lesen des Titels (Befehls-Ausgabe)
   if (! $args -> {"rttitle"}) {
      $pth = catfile ($args -> {"kaldir"}, $args -> {"ixfile"});
      $pth .= ".de" unless -f $pth;
      if ( -f $pth ) {
```

```
         $cd = replace ($args -> {"titlecmd"}, { "file" => $pth, });
         $ch = undef;
         open ($ch, "$cd |") or do {
            print STDERR "Kann Befehl $cd nicht ausführen $!\n";
            exit;
         };
         binmode ($ch, ":encoding(utf-8)");
         $args -> {"rttitle"} = <$ch>;
         close ($ch);
      }
   }
   $args -> {"rttitle"} or $args -> {"rttitle"} = "Kalender";
} # set_default


# gibt die Version nach STDOUT aus
sub version {
   print << 'VERSION';
KLEIDER/web/src/kalender/tree.pl
Baumstruktur der Kalender
2015-12-22
2015 - 2020 Herbert Schiemann <h.schiemann@herbaer.de>
VERSION
};
$args -> {"[sr]version"} = sub { version (); exit 0; };


$args -> {"[sr]help"} = sub {
   set_default ($args);
   version ();
   print_message_with_values (<<"HELP", $args);
$0 --help     zeigt diese Hilfe an
$0 --version  zeigt die Programm-Version an

$0 [option]...
--[no_]verbose            Mehr Meldungen nach STDERR \${[cnt]verbose}
--lang       LANG         Kennung der Sprache \${lang}
--webdir     WEBDIR       Basisverzeichnis zur Website
                          \${webdir}
--kaldir     KALDIR       Verzeichnis der Kalender-Quellen
                          \${kaldir}
--shiftnd    SHIFTND ..   Verschobene Knoten \${shiftnd}
--incldir    INCLDIR ..   Regex der eingeschlossenen Verzeichnisse \${incldir}
--excldir    EXCLDIR ..   Regex der ausgenommenen Verzeichnisse \${excldir}
--inclfile   INCLFILE ..  Regex der eingeschlossenen Dateien \${inclfile}
--exclfile   EXCLFILE ..  Regex der ausgenommenen Dateien \${exclfile}
--rtname     RTNAME       Name des Wurzelknotens \${rtname}
--rtrefbase  RTREFBASE    Verweis-Basis des Wurzelknotens \${rtrefbase}
--rttitle    RTTITLE      Titel des Wurzelknotens \${rttitle}
--ixfile     IXFILE       Default-Index-Datei \${ixfile}
--dirnmflt   DIRNMFLT ..  Namensfilter für Verzeichnisse \${dirnmflt}
--filenmflt  FILENMFLT .. Namensfilter für Dateien \${filenmflt}
--titlecmd   TITLECMD     Befehl zur Bestimmung der Titels mit Platzhaltern
                          \${titlecmd}
--xmlns      XMLNS        XML-Namensraum der Ausgabe
                          \${xmlns}
HELP
   exit 0;
}; # help


read_args   ($args);
set_default ($args);


collect_data ($args, $data);
add_shiftnd  ($args, $data);
print_data   ($args, $data);


sub collect_data {
   my ($args, $data) = @_;
   my $node  = {
      "name"     => $args -> {"rtname"},
      "refbase"  => $args -> {"rtrefbase"},
      "title"    => $args -> {"rttitle"},
      "ref"      => $args -> {"ixfile"},
      "node"     => [],
   } ;
   $data -> {"node"} = [$node];
   $args -> {"rid"} = [ map { qr/$_/ } @{$args -> {"incldir"}}   ];
   $args -> {"red"} = [ map { qr/$_/ } @{$args -> {"excldir"}}   ];
   $args -> {"rif"} = [ map { qr/$_/ } @{$args -> {"inclfile"}}  ];
   $args -> {"ref"} = [ map { qr/$_/ } @{$args -> {"exclfile"}}  ];
   $args -> {"dnf"} = [ map { qr/$_/ } @{$args -> {"dirnmflt"}}  ];
   $args -> {"fnf"} = [ map { qr/$_/ } @{$args -> {"filenmflt"}} ];
   process_dir ($args, $args -> {"kaldir"}, $node -> {"node"});
} # collect_data
```

```
# Die Daten eines Verzeichnisses "einsammeln"
sub process_dir {
   my ($args, $dir, $nodes) = @_;
   my $dh;   # Verzeichnis-Handle
   my $de;   # Verzeichniseintrag
   my $path; # Dateipfad
   my $verb = $args -> {"[cnt]verbose"};
   opendir ($dh, $dir) or do {
      print STDERR "Kann Verzeichnis $dir nicht öffnen: $!\n" if $verb;
      return;
   };
   my $re;      # ein regulärer Ausdruck
   my $n;       # ein Zähler
   my $rid = $args -> {"rid"};
   my $red = $args -> {"red"};
   my $rif = $args -> {"rif"};
   my $ref = $args -> {"ref"};
   my $dnf = $args -> {"dnf"};
   my $fnf = $args -> {"fnf"};
   my $cmd = $args -> {"titlecmd"};
   my $pth;      # Dateipfad einer Index-Quelldatei
   my $cd;       # Befehl nach der Ersetzung der Platzhalter
   my $ch;       # Handle zum Lesen der Ausgabe des Befehls

   # Knotendaten
   my $node;     # ein Knoten
   my $name;     # Name
   my $refbase;  # Verweis-Basis
   my $title;    # Titel
   my $link;     # Verweis
   my $repl = { "file" => undef, }; # Werte der Platzhalter in titlecmd

   while (defined ($de = readdir ($dh))) {
      next if $de eq "." || $de eq ".." ;
      $path = catfile ($dir, $de);
      if (-f $path) {
         $n = 0;
         for $re (@$rif) {
            if ( $path =~ $re ) { ++$n; last; }
         }
         next unless $n;
         $n = 0;
         for $re (@$ref) {
            if ( $path =~ $re ) { ++$n; last; }
         }
         next if $n;
         print STDERR "file $path\n" if $verb;
         $name = $de;
         for $re (@$fnf) {
            if ( $path =~ $re ) { $name = $1; last; }
         }
         $de =~ s/\.$//;
         $node = {
            "ref"  => $de,
            "name" => $name,
         };
         push (@$nodes, $node);
         $repl -> {"file"} = $path;
         $cd = replace ($cmd, $repl);
         $ch = undef;
         open ($ch, "$cd |") or do {
            print STDERR "Kann Befehl $cd nicht ausführen $!\n";
            exit;
         };
         binmode ($ch, ":encoding(utf-8)");
         $title = <$ch>;
         close ($ch);
         $node -> {"title"} = $title if $title;
      }
      elsif (-d $path) {
         $n = 0;
         for $re (@$rid) {
            if ( $path =~ $re ) { ++$n; last; }
         }
         next unless $n;
         $n = 0;
         for $re (@$red) {
            if ( $path =~ $re ) { ++$n; last; }
         }
         next if $n;
         print STDERR "dir $path\n" if $verb;
         $name = $de;
         for $re (@$dnf) {
            if ( $path =~ $re ) { $name = $1; last; }
         }
         $node = {
            "refbase" => $de,
            "name"    => $name,
            "ref"     => $args -> {"ixfile"},
            "node"    => [],
         };
         $pth = catfile ($path, $args -> {"ixfile"});
         $pth .= ".de" unless -f $pth;
         if ( -f $pth ) {
            $cd = replace ($args -> {"titlecmd"}, { "file" => $pth, });
            $ch = undef;
```

```
            open ($ch, "$cd |") or do {
               print STDERR "Kann Befehl $cd nicht ausführen $!\n";
               exit;
            };
            binmode ($ch, ":encoding(utf-8)");
            $node -> {"title"} = <$ch>;
            close ($ch);
         }
         $node -> {"title"} ||= $name;
         push (@$nodes, $node);
         process_dir ($args, $path, $node -> {"node"});
      }
   }
   closedir ($dh);
} # process_dir


# Den Inhalt der "Schiebeknoten" oben in der Baumansicht hinzufügen
sub add_shiftnd {
   my ($args, $data) = @_;
   my $nodes = $data -> {"node"} -> [0] -> {"node"};
   return unless $nodes;
   my $sn = {};
   my $nm;
   for $nm (@{$args -> {"shiftnd"}}) {
      ++$sn -> {$nm};
   }
   my $nn = [];     # neue Knoten
   my $n;           # ein Knoten
   my $c;           # Kindknoten
   my $nl;          # Knotenliste
   my $prf;         # Verweis-Präfix
   my $cc;          # neuer Knoten
   for $n (@$nodes) {
      if ($sn -> {$n -> {"name"}}) {
         $nl = $n -> {"node"} or next;
         $prf = $n -> {"refbase"} || "";
         $prf .= "/" if $prf;
         for $c (@$nl) {
            $cc = {
               "name"  => $c -> {"name"},
               "title" => $c -> {"title"},
               "ref"   => $prf . $c -> {"ref"}
            };
            $cc -> {"refbase"} = $prf . $c -> {"refbase"} if $c -> {"refbase"};
            $cc -> {"node"}    = $c -> {"node"}           if $c -> {"node"};
            push (@$nn, $cc);
         }
      }
   }
   unshift (@$nodes, @$nn);
} # add_shiftnd


sub print_data {
   my ($args, $data) = @_;
   my $opt = {};
   my $writer = Herbaer::XMLDataWriter -> new ($opt);
   $writer -> open ("-", "utf-8", $args -> {"xmlns"}, "");
   $writer -> write ("tree", {}, $data);
   $writer -> close ();
} # print_data
```

# titlecmd

[Quelltext]

# Übersicht

`titlecmd.pl --help|--version`

`titlecmd.pl` *`FILE`* [*`LANG`*]

# Beschreibung

Dieses Skript wird vom Programm `tree.pl` aufgerufen. Es gibt den Titel des XML-Dokuments unter dem Dateipfad *`FILE`* in der Sprache *`LANG`* aus.

Die XSLT-Transformation `xmlnsss.xslt` (beschrieben im Zusammenhang mit der Lokalisierung / Übersetzung) liefert den XML-Namensraum des Wurzelelements und den Basisnamen der Transformation aus der `xml-stylesheet`-Verarbeitungsanweisung.

Für XHTML-Dokumente liefert die Transformation `xhtml_title.xslt` den Titel, für Kalenderbilder die Transformation `kalender_title.xslt`.

# Quelltext

[Beschreibung]

```
#!/bin/bash
# Befehl zur Bestimmung des Titels einer Datei
# 2016-01-17 Herbert Schiemann <h.schiemann@herbaer.de>
# usage:
# titlecmd FILE LANG
# von tree.pl aufgerufen


argcnt=$#       ; # Zahl der Argumente
if (( argcnt == 0 )) || [[ "$1" == "--help" ]] || [[ "$1" == "--version" ]]; then
   cat <<EOF ;
KLEDER/web/src/kalender/titlecmd
Titel eines Dokuments/Datei
2016-01-17 Herbert Schiemann <h.schiemann@herbaer.de>
usage:
kalender/titlecmd FILEPATH LANGUAGE
EOF
   exit 0;
fi;


srcdir=$(realpath $0)
srcdir=${srcdir%/*} ;  # Skript-verzeichnis
xmlnsss=${srcdir%/*}/localization/xmlnsss.xslt;
[[ -f $xmlnsss  ]] || exit;
file=$1;
[[ -f $file ]]     || exit;
lang=de ;
(( argcnt > 1 )) && lang=$2 ;
nsss=$(xsltproc "$xmlnsss" "$file"); # XML-Namensraum | Stylesheet-Basisname
ns=${nsss%%|*};
if [[ "$ns" == "http://herbaer.de/xmlns/20151211/kalenderbilder/" ]]; then
   tt=$srcdir/kalender_title.xslt;
   [[ -f $tt ]] || exit;
   xsltproc --stringparam p_lang $lang $tt $file ;
elif [[ "$ns" == "http://www.w3.org/1999/xhtml" ]]; then
   tt=$srcdir/xhtml_title.xslt;
   [[ -f $tt ]] || exit;
   xsltproc --stringparam p_lang $lang $tt $file ;
fi;
```

# xhtml_title.xslt

[Quelltext]

## Namensräume

| Präfix | Namensraum |
|---|---|
| xml | http://www.w3.org/XML/1998/namespace |
| d | http://herbaer.de/xmlns/20051201/doc |
| ht | http://www.w3.org/1999/xhtml |
| xsl | http://www.w3.org/1999/XSL/Transform |

## Ausgabe (output)

| | |
|---|---|
| Method | text |
| Encoding | utf-8 |

## Parameter

### Parameter p_lang

Die gewünschte Sprache des Titels: Zuerst wird ein Titel in der gewünschte Sprache einschließlich der Land-Kennung gesucht. dann ein Titel in der gewünschten Sprache ohne die Land-Kennung, zuletzt der erste Titel

Select: 'de'

Der Parameter wird in den folgenden Toplevel-Elementen benutzt:

Muster-Vorlage ht:head | * [ht:title]

## Muster-Vorlagen (matching templates)

### Muster-Vorlage /

Wurzel

### Muster-Vorlage *

Elemente werden absteigend durchsucht. Falls ein `html`- oder `head`-Element als Kind enthalten ist, wird nur das erste dieser Elemente verarbeitet.

### Muster-Vorlage ht:head | * [ht:title]

HTML-Kopf: der erste Titel in der passenden Sprache mit oder ohne Land-Kennung

Verwendete globale Parameter oder Variable:

Parameter p_lang

# Quelltext

[Beschreibung]

```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet href="/pool/xslt_ht.xslt" type="application/xml"?>
<!--
  Titel eines XHTML-Dokuments
  2016 Herbert Schiemann <h.schiemann@herbaer.de>
  Borkener Str. 167, 46284 Dorsten, Germany
  GPL Version 2 oder neuer
  Jede Gewährleistung ist ausgeschlossen
-->
<xsl:stylesheet
  xmlns:xsl = "http://www.w3.org/1999/XSL/Transform"
  xmlns:ht  = "http://www.w3.org/1999/xhtml"
  xmlns:d   = "http://herbaer.de/xmlns/20051201/doc"
  version   = "1.0"
>
<xsl:param name = "p_lang" select  = "'de'"/>


<xsl:output method = "text" encoding = "utf-8"/>


<xsl:template match = "/">
  <xsl:apply-templates select = "*"/>
</xsl:template>


<xsl:template match = "*">
  <xsl:choose>
    <xsl:when test = "ht:html">
      <xsl:apply-templates select = "ht:html [1]"/>
    </xsl:when>
    <xsl:when test = "ht:head">
      <xsl:apply-templates select = "ht:head [1]"/>
    </xsl:when>
    <xsl:otherwise>
      <xsl:apply-templates select = "*"/>
    </xsl:otherwise>
  </xsl:choose>
</xsl:template>


<xsl:template match = "ht:head | * [ht:title] ">
  <xsl:variable name = "l2">
    <xsl:choose>
      <xsl:when test = "contains ($p_lang, '-')">
        <xsl:value-of select = "substring-before ($p_lang, '-')"/>
      </xsl:when>
      <xsl:otherwise>
        <xsl:value-of select = "$p_lang"/>
      </xsl:otherwise>
    </xsl:choose>
  </xsl:variable>
  <xsl:choose>
    <xsl:when test = "ht:title [@lang = $p_lang or @xml:lang = $p_lang]">
      <xsl:value-of select = "ht:title [@lang = $p_lang or @xml:lang = $p_lang][1]"/>
    </xsl:when>
    <xsl:when test = "ht:title [@lang = $l2 or @xml:lang = $l2]">
      <xsl:value-of select = "ht:title [@lang = $l2 or @xml:lang = $l2][1]"/>
    </xsl:when>
    <xsl:otherwise>
      <xsl:value-of select = "ht:title"/>
    </xsl:otherwise>
  </xsl:choose>
</xsl:template>


</xsl:stylesheet>
```

# kalender_title.xslt

[Quelltext]

## Namensräume

| Präfix | Namensraum |
|---|---|
| xml | http://www.w3.org/XML/1998/namespace |
| lt | http://herbaer.de/xmlns/20151212/loctext/ |
| kb | http://herbaer.de/xmlns/20151211/kalenderbilder/ |
| d | http://herbaer.de/xmlns/20051201/doc |
| xsl | http://www.w3.org/1999/XSL/Transform |

## Ausgabe (output)

| | |
|---|---|
| Method | text |
| Encoding | utf-8 |

## Parameter

### Parameter p_lang

Die gewünschte Sprache des Titels. Sie kann eine Kennung des Landes enthalten, z.B. `zh-cn`.

Zunächst wird ein Titel für die gewünschte Sprach-Land-Kombination gesucht.
Wenn es den nicht gibt, wird der Titel in der gewünschten Sprache ohne die Kennung des Landes gesucht.
Wenn es den auch nicht gibt, wird der Titel in der ersten angegebenen Sprache ausgegeben.
Wenn es gar keinen Titel gibt, wird das Wort `Kalender` ausgegeben.

Select: 'de'

Der Parameter wird in den folgenden Toplevel-Elementen benutzt:

Muster-Vorlage /kb:kalenderbilder

### Parameter p_year

Mit Jahreszahl? (0 nein, 1 ja)

Wenn dieser Parameter `1` ist und eine Jahreszahl (`kb:y`) zu der Bildauswahl angegeben ist, dann wird die Jahreszahl an den Titel angehängt, falls er nicht schon die Jahreszahl enthält.

Select: '0'

Der Parameter wird in den folgenden Toplevel-Elementen benutzt:

Muster-Vorlage /kb:kalenderbilder

## Muster-Vorlagen (matching templates)

### Muster-Vorlage /kb:kalenderbilder

Der angegebene Titel in der gewünschten Sprache oder in der ersten angegebenen Sprache

Verwendete globale Parameter oder Variable:

Parameter p_lang

Parameter p_year

## Muster-Vorlage kb:y

### Parameter

t

Das Kalenderjahr, wenn es nicht bereits im Titel angegeben ist

# Quelltext

[Beschreibung]

```
<?xml version="1.0" encoding="utf-8"?>
<?xml-stylesheet href="/pool/xslt_ht.xslt" type="application/xml"?>
<!--
  Titel eines Kalenders
  2015 Herbert Schiemann <h.schiemann@herbaer.de>
  Borkener Str. 167, 46284 Dorsten, Germany
  GPL Version 2 oder neur
  Jede Gewährleistung ist ausgeschlossen
-->
<xsl:stylesheet
  xmlns:xsl = "http://www.w3.org/1999/XSL/Transform"
  xmlns:d   = "http://herbaer.de/xmlns/20051201/doc"
  xmlns:kb  = "http://herbaer.de/xmlns/20151211/kalenderbilder/"
  xmlns:lt = "http://herbaer.de/xmlns/20151212/loctext/"
  version   = "1.0"
>
<xsl:param name = "p_lang" select  = "'de'"/>


<xsl:param name = "p_year" select = "'0'"/>


<xsl:output method = "text" encoding = "utf-8"/>


<xsl:template match = "/kb:kalenderbilder">
  <xsl:variable name = "l2">
    <xsl:choose>
      <xsl:when test = "contains ($p_lang, '-')">
        <xsl:value-of select = "substring-before ($p_lang, '-')"/>
      </xsl:when>
      <xsl:otherwise>
        <xsl:value-of select = "$p_lang"/>
      </xsl:otherwise>
    </xsl:choose>
  </xsl:variable>
  <xsl:variable name = "t">
    <xsl:choose>
      <xsl:when test = "kb:t/lt:v/lt:t [@l = $p_lang]">
        <xsl:value-of select = "kb:t/lt:v/lt:t [@l = $p_lang]"/>
      </xsl:when>
      <xsl:when test = "kb:t/lt:v/lt:t [@l = $l2]">
        <xsl:value-of select = "kb:t/lt:v/lt:t [@l = $l2]"/>
      </xsl:when>
      <xsl:when test = "kb:t/lt:v/lt:t">
        <xsl:value-of select = "kb:t/lt:v/lt:t [1]"/>
      </xsl:when>
      <xsl:when test = "kb:t">
        <xsl:value-of select = "kb:t"/>
      </xsl:when>
      <xsl:otherwise>
        <xsl:text>Kalender</xsl:text>
      </xsl:otherwise>
    </xsl:choose>
  </xsl:variable>
  <xsl:value-of select = "$t"/>
  <xsl:if test = "$p_year = '1'">
    <xsl:apply-templates select = "kb:y">
      <xsl:with-param name = "t" select = "$t"/>
    </xsl:apply-templates>
  </xsl:if>
</xsl:template>


<xsl:template match = "kb:y">
  <xsl:param name = "t"/>
  <xsl:if test = "not (contains ($t, .))">
    <xsl:value-of select = "concat (' ', .)"/>
  </xsl:if>
</xsl:template>


</xsl:stylesheet>
```

# Herbaer::Readargs

[Quelltext]

# Übersicht

```
use Herbaer::Readargs;
my $args = {
   "[cnt]verbose" => 1,
   "param"        => "auto",
   "name"         => [],
   "option"       => undef,
   "[sr]help"     => /help,
};
sub help {
  print_message_with_values ('param ${param}', $args);
  exit 0;
};
read_args ($args);
```

# Beschreibung

Das Perl-Modul `Herbaer::Readargs` bietet die Funktionen

- `read_args`
- `print_message_with_values`
- `read_data_file`

In der Voreinstellung werden die Funktionen `read_args` und `print_message_with_values` exportiert.

# `read_args` (*`$args`*, *`$argv`*)

## Parameter *`$argv`*

Der Parameter *`$argv`* kann ein Array von Zeichenketten referenzieren. In diesem Fall werden die Einträge des referenzierten Arrays statt der Befehlszeilen-Argumente @ARGV verarbeitet. Andernfalls wird *`$argv`* als Referenz auf @ARGV initialisiert.

## Parameter *`$args`*

Der Parameter *`$args`* ist eine HASH-Referenz; wenn nicht, dann endet das Unterprogramm sofort. Die Schlüssel und vorbelegten Werte bestimmen die erlaubten Argumente. Die Funktion trägt die tatsächlichen Werte aus den Aufruf-Argumenten in den HASH ein.

Die Schlüssel des HASH haben die Form `[`*`TYPE`*`]`*`NAME`* oder *`NAME`*. *`TYPE`* bezeichnet den Typ eines Arguments. *`NAME`* steht für eine Folge von Buchstaben, Ziffern oder dem Unterstrich ("_"). Der Name der Option ergibt sich, indem zwei Minus-Zeichen vorangestellt werden: `--`*`NAME`*.

## Schlüssel `[cnt]`*`NAME`*

Der Wert zu dem Schlüssel ist ein Zähler (nicht-negative ganze Zahl). Jede Option `--`*`NAME`* erhöht den Zähler um eins oder setzt den Wert auf eins, falls er noch nicht definiert ist. Jede Option `--no_`*`NAME`* setzt den Zähler auf den Wert null.

Eine typische Anwendung ist der Schlüssel `[cnt]verbose`. Der Wert steuert üblicherweise den Umfang der Ausgabe an STDERR.

## Schlüssel `[sr]`*`NAME`*

Der voreingestellte Wert zu diesem Schlüssel muß ein Unterprogramm referenzieren. Die Option `--`*`NAME`* führt zum Aufruf des Unterprogramms mit den Parametern *`$args`* und *`$argv`*.

Typische Anwendungen sind die Schlüssel `[sr]help` (Option `--help`) und `[sr]version` (Option `--version`).

## Schlüssel `[rc]`*`NAME`*

Das Argument `--`*`NAME`* dient dazu, weitere Daten aus einer Datei zu lesen. Das folgende Argument *`FILE`* bestimmt die Datei. Es wird die Funktion `read_data_file ("`*`FILE`*`", $args, "[rc]`*`NAME`*`")` aufgerufen.

## Andere Schlüssel mit einer ARRAY-Referenz als Wert

Alle anderen Schlüssel werden abhängig davon behandelt, ob der voreingestellte Wert ein ARRAY referenziert oder nicht.

Wenn der voreingestellte Werte eine ARRAY-Referenz ist, werden das erste folgenden Argument nach der Option `--`*`NAME`* und die weiteren folgenden Argumente, solange sie nicht mit "`--`" anfangen, als Zeichenketten an das referenzierte Array angehängt. Das erste Argument nach `--`*`NAME`* wird stets als Wert an das Array angehängt, auch wenn es mit "`--`" anfängt.

## Andere Schlüssel ohne Wert oder mit einfachem Wert

Das Argument, das der Option `--`*`NAME`* folgt, wird als Wert in den HASH eingetragen. Ein existierender Wert wird ersetzt.

## Positionsargumente

Das erste Argument, das nicht Wert zu einer Option ist und nicht dem Muster `--`*`NAME`* einer Option entspricht, wird als Positionsargument interpretiert. Wenn der Hash-Schlüssel `_argv` existiert und sein Wert ein Array referenziert, werden das Argument und alle folgenden Argumente als Zeichenketten an das Array angehängt. Andernfalls wird ein Hash-Eintrag zu dem Schlüssel `_argv` neu angelegt mit einer Referenz auf ein leeres Array als Wert. Die Positionsargumente werden an das Array angehängt.

Das Argument `--` hat eine spezielle Funktion: es ist selbst kein Positions-Argument, aber alle folgenden Argumente sind Positions-Argumente.

Wenn ein Argument weder Wert einer Option noch Positionsargument ist und dem Muster `--`*`NAME`* einer Option entspricht, aber der Hash keinen passenden Schlüssel enthält, bricht das Programm mit dem Ende-Code 1 ab. Wenn der Zähler `[cnt]verbose` definiert und nicht Null ist, wird eine Meldung an STDERR ausgegeben.

# `print_message_with_values` (*`$msg`*, *`$vals`*)

Ersetzt Platzhalter im Text *`$msg`* und gibt den Text nach STDOUT aus. Diese Funktion erleichtert die Ausgabe einer Hilfe mit Angabe der aktuellen Einstellungen.

*`$msg`*

Der Text der Meldung mit Platzhaltern. Die Platzhalter haben die Form `${`*`NAME`*`}`. Teilzeichenketten der Form `$<{`*`NAME`*`>}` werden durch `${`*`NAME`*`}` ersetzt.

Wenn `$vals -> {"`*`NAME`*`"}` nicht definiert ist, wird der Platzhalter entfernt.

Wenn `$vals -> {"`*`NAME`*`"}` ein ARRAY referenziert, wird der Platzhalter durch eine komma-getrennte Liste der Werte des ARRAY in eckigen Klammern ersetzt.

Andernfalls wird der Platzhalter durch den Wert `$vals -> {"`*`NAME`*`"}` in Klammern ersetzt.

*`$vals`*

Eine Referenz auf einen HASH mit den Werten, für die die Platzhalter stehen. In der Regel ist *`$vals`* das Ergebnis der Funktion `read_args`.

# `read_data_file` (*`$file`*, *`$args`*, *`$k`*)

Trägt Daten aus einer Datei in den Hash *`$args`* ein. Der Rückgabewert ist der Hash *`$args`*.

*`$file`*

Wenn *`$file`* ein absoluter Pfad ist, wird die Datei unter diesem Pfad gelesen. Wenn andernfalls *`$file`* nicht mit `.rc` endet, wird `.rc` an *`$file`* angehängt. Die Datei wird unter dem aktuellen Verzeichnis `.` und den Verzeichnissen `$HOME/etc` und `/etc` gesucht. Die erste gefundene Datei wird gelesen. Wenn die Datei nicht gefunden wird, endet die Funktion.

*`$args`*

*`$args`* referenziert den Hash, in den die gelesenen Daten eingetragen werden. Wenn *`$args`* keine Hash-Referenz ist, wird der Hash neu erzeugt.

Der Wert unter dem Schlüssel `[cnt]verbose` bestimmt den Umfang der Meldungen nach STDERR.

*`$k`*

Wenn der Wert von *`$k`* definiert ist, wird der Dateipfad der gelesenen Datei unter diesem Schlüssel in den Hash *`$args`* eingetragen. Wenn *`$file`* ein absoluter Dateipfad ist, wird dieser Pfad eingetragen, auch wenn die Datei nicht gelesen werden kann.

## Dateneinträge

Die Daten-Datei enthält Text in UTF-8-Kodierung. Ein Zeilenanfang der Form *`KEY = VAL`* leitet einen Dateneintrag ein. *`KEY`* ist eine nicht-leere Folge von Buchstaben, Ziffern, Unterstrichen und Punkten, die vorn und hinten von beliebig vielen Leerzeichen umgeben sein kann. Auch das Gleichheitszeichen kann von beliebig vielen Leerzeichen umgeben sein. *`VAL`* ist der Anfang eines Wertes und beginnt mit einem Nicht-Leerzeichen.

Der Wert, den die mit *`VAL`* beginnende Zeichenkette darstellt, kann ein „nacktes” Wort, eine angeführte Zeichenkette, ein Array oder ein Hash sein. Die Darstellung des Wertes kann mehrere Zeilen umfassen. In der Zeile, in der der Wert endet, kann noch ein Kommentar folgen.

Wenn der Schlüssel `[cnt]`*`KEY`* im Hash *`$args`* existiert, wird der Wert unter diesem Schlüssel eingetragen, sonst unter dem Schlüssel *`KEY`*.

## „Nackte” Wörter

Ein „nacktes” Wort beginnt mit einem Nicht-Leerzeichen und endet vor dem ersten folgenden Leerzeichen, der ersten folgenden geschweiften oder eckigen Klammer, dem ersten folgenden Komma oder Gleichheitszeichen oder natürlich dem Ende der Datei.

## Angeführte Zeichenketten

Eine angeführte Zeichenkette wird mit dem Zeichen `"` eingeleitet. In den folgenden Zeichen steht die Zeichenfolge `\ \` für das einzelne Zeichen \, die Zeichenfolge `\"` für das einzelne Zeichen ", jedes andere Zeichen als `"` steht für sich selbst. Der Wert der angeführten Zeichenkette endet vor dem nächsten "ungeschützten" Anführungszeichen `"`. Das schließende Anführungszeichen gehört zur Darstellung des Wertes.

## Arrays

Die Darstellung eines Arrays beginnt mit dem Zeichen `[` und endet mit dem Zeichen `]`. Die Werte des Array können von Leerzeichen und beliebig vielen Kommata und Kommentaren umgeben sein. Ein „nacktes” Wort als Wert eines Arrays kann daher effektiv nicht mit einem Komma oder dem Zeichen # beginnen.

Ein Wert eines Arrays kann selbst ein Array oder ein Hash sein.

## Hash

Die Darstellung eines Hash beginnt mit dem Zeichen `{` und endet mit dem Zeichen `}`. Die Einträge haben die Form *KEY* `=>` *VALUE*. *KEY* steht für ein „nacktes” Wort oder eine angeführte Zeichenkette als Schlüssel, *VALUE* für einen beliebigen Wert („nacktes” Wort, angeführte Zeichenette, Array oder Hash), unter dem Schlüssel *KEY*.

*KEY* und *VALUE* können von Kommentaren umgeben sein. Ein nacktes Wort als Schlüssel oder Wert kann daher nicht mit dem Zeichen # beginnen. Statt der beiden Zeichen `=>` kann auch ein einzelnes Gleichheitszeichen stehen.

Die Einträge eines Hash (Schlüssel-Wert-Paare) können von Leerzeichen, beliebig vielen Kommata und Kommentaren umgeben sein. Ein nacktes Wort als Schlüssel kann daher auch nicht mit einem Komma beginnen.

## Kommentare

Ein Kommentar beginnt mit dem Zeichen #, das nicht Teil eines nackten Wortes oder einer angeführten Zeichenkette ist, und endet mit dem folgenden Zeilenende.

# Installation

Die Datei muss unter dem relativen Pfad *INC*`/Herbaer/Readargs.pm` gefunden weren. *INC* steht für einen Eintrag in der Liste `@INC`. Auf meinem System ist der Pfad der Datei `/usr/local/share/perl/5.10.1/Herbaer/Readargs.pm`.

# Quelltext

[Beschreibung]

```
# Liest die (Befehlszeilen-)Argumente
# 2016-06-16 Herbert Schiemann <h.schiemann@herbaer.de>
# GPL Version 2 oder neuer


# 2020-04-18 print_message_with_values: bessere Ausgabe von ARRAY-Werten


package Herbaer::Readargs;


BEGIN {
   use File::Spec::Functions qw (file_name_is_absolute catfile);
   use Exporter;
   our $VERSION    = 20160617;
   our @ISA        = qw (Exporter);
   our @EXPORT     = qw (read_args print_message_with_values);
   our @EXPORT_OK  = qw (read_args print_message_with_values read_data_file);
}


my $verb;
use utf8;
binmode (STDERR, "raw:encoding(utf8)");


# verarbeitet die Argumente
# Die gültigen Options-Namen sind in $args voreingestellt
# read_args ($args);
sub read_args {
   my ($args, $argv) = @_;
   ref ($args) eq "HASH"  or return;
   ref ($argv) eq "ARRAY" or $argv = \@ARGV;
   my $a;                  # ein einzelnes Argument in der Schleife
   my $oname;              # Name einer Option
   my $okey;               # Schlüssel des Hash-Eintrags zur letzten Option
   my $ok;                 # Test-Schlüssel
   my $kfound;             # neuer Schlüssel gefunden?
   my $otypes = [          # möglicher Typen von Optionen
      "",                  # einfacher Wert (Zeichenkette)
      "[cnt]",             # logischer Wert (durch --no_XXX zurückgesetzt)
      "[sr]",              # Unterprogramm-Referenz, Parameter $args, $argv
      "[rc]",              # Name oder Pfad einer Einstellungsdatei
   ];
   my $ot;                 # Options-Typ (s. $otypes)
   my $ott;                # Options-Typ in Schleife
   my $expect_val = 0;     # erwarte einen Wert
   my $rd_posargs = 0;     # lese Positions-Argumente $args -> {'_argv'}
   my $car;                # aktuelle Liste ("current array")
   for $a (@$argv) {
      if ($rd_posargs) { push (@$car, $a); next; }
      if ( !$expect_val && $a =~ /^--([a-zA-Z0-9_]*)/ ) {
         $oname = $1;
         if ($oname eq '') {
            ref ($car = $args -> {'_argv'}) eq "ARRAY"
                or $args -> {'_argv'} = $car = []
                ;
            $rd_posargs = 1;
            next;
         }
         $kfound = 0;
         for $ott (@$otypes) {
            $ok = "$ott$oname";
            if (exists $args -> {$ok}) {
               $ot   = $ott;
               $okey = $ok;
               ++ $kfound;
               ++ $args -> {$ok} if $ott eq '[cnt]';
            }
            if ( $ott eq '[cnt]' && $oname =~ /^no_(.+)/ ){
               $ok = "$ott$1";
               if (exists $args -> {$ok}) {$ot = $ott; ++ $kfound; $args -> {$ok} = 0; }
            }
            last if $kfound;
         }
         if ($kfound) {
            $car = undef;
            if    ($ot eq '[cnt]') {}
            elsif ($ot eq '[sr]')  { $args -> {$okey} -> ($args, $argv); }
            else {
               ++ $expect_val;
               ref ($car = $args -> {$okey}) eq "ARRAY" or $car = undef;
            }
            next;
         }
         elsif (!$car) {
            print STDERR "Ungültige Option $a\n" if $args -> {"[cnt]verbose"};
            exit 1;
         }
```

```
      }
      if ($car) { push (@$car, $a); }
      elsif ($okey && $expect_val) {
         if ($ot eq "[rc]") {
            read_data_file ($a, $args, $okey);
         }
         else {
            $args -> {$okey} = $a;
         }
      }
      else {
         ref ($car = $args -> {'_argv'}) eq "ARRAY"
             or $args -> {'_argv'} = $car = []
             ;
         push (@$car, $a);
         $rd_posargs = 1;
      }
      $okey = undef;
      $expect_val = 0;
   }
   $args;
} # read_args


sub print_message_with_values {
   my ($msg, $vals) = @_;
   my $replace = sub {
      my ($s, $k) = @_ ;
      my $v = $vals -> {$k};
      $k =~ /^<(.*)>$/    ? "$s\${$1}"                        :
      ! defined ($v)      ? $s                                :
      ref ($v) eq "ARRAY" ? "$s\[" . join (",$s", @$v) . "\]" :
                            "$s(" . $v . ")"                  ;
   }; # replace
   $msg =~ s/(\s*)\$\{(.*?)\}/$replace -> ($1, $2)/ges ;
   print $msg;
} # help


# liest eine Daten-Datei
sub read_data_file {
   my ($file, $args, $k) = @_;
   ref ($args) eq "HASH" or $args = {};
   $verb = $args -> {"[cnt]verbose"};
   my $f; # Dateipfad
   my $d; # Verzeichnis
   my $h; # Dateihandle
   if (file_name_is_absolute ($file)) {
      $f = $file;
   }
   else {
      $file =~ /\.rc$/ or $file .= ".rc";
      for $d (".", "$ENV{HOME}/etc", "/etc") {
         $f = catfile ($d, $file);
         last if -f $f;
      }
   }
   if (!$f) {
      print STDERR "Datei nicht angegeben.\n" if $verb;
      return $args;
   }
   $args -> {$k} = $f if $k;
   if ( !open ($h, "<:encoding(utf-8)", $f)) {
      print STDERR "Kann Datei \"$f\" nicht lesen: $!\n" if $verb;
   }
   else {
      print STDERR "Lese Datei \"$f\"\n" if $verb;
      my $line;
      my $key;
      my $val;
      while ($h && (defined ($line = <$h>))) {
         print STDERR $line if $verb > 2;
         next unless $line =~ s/^\s*([a-zA-Z0-9_.]+)\s*=\s*//;
         $key = $1;
         next unless $line;
         if (exists $args -> {"[cnt]$key"}) {
            $key = "[cnt]$key";
         }
         if (defined ($v = _read_value (\$line, $h))) {
            print STDERR "key/value $key $v\n" if $verb > 2;
            $args -> {$key} = $v;
         }
         $line =~ s/\s+//;
         $line =~ s/#.*//s;
         print STDERR "unerwarteter Zeilenrest $line\n" if $line && $verb;
      }
   }
   $args;
} # read_data_file
```

```
sub _read_value {
   my ($lref, $h) = @_;
   my $c;
   print STDERR "_read_value line:\"" . $$lref . "\"\n" if $verb > 2;
   $$lref =~ s/(.)// or return undef;
   $c = $1;
   my $v = undef;
   my $k;
   my $v2;
   if ($c eq "\[") {
      $v = [];
      print STDERR "neues Array\n" if $verb > 2;
      while (1) {
         _skip_comment ($lref, $h);
         next if $$lref =~ s/^,//;
         if ( $$lref =~ s/^]// ) {
            my $n = @$v;
            print STDERR "Array abgeschlossen, $n Elemente\n" if $verb > 2;
            last;
         }
         if (!defined ($v2 = _read_value ($lref, $h))) {
            print STDERR "fehlender Wert\n" if $verb;
            last;
         }
         else {
            print STDERR "Array-Element $v2\n" if $verb > 2;
            push (@$v, $v2);
         }
      }
   }
   elsif ($c eq "\{") {
      $v = {};
      print STDERR "neuer Hash\n" if $verb > 2;
      while (1) {
         _skip_comment ($lref, $h);
         next if $$lref =~ s/^,//;
         if ( $$lref =~ s/^}// ) {
            print STDERR "Hash abgeschlossen\n" if $verb > 2;
            last;
         }
         if (! defined ($k = _read_value ($lref, $h))) {
            print STDERR "Hash-Schlüssel erwartet\n" if $verb > 2;
            last;
         }
         if (ref ($k)) {
            print STDERR
               "Skalarer Wert als Schlüssel erwartet, nicht " . ref ($k) . "\n"
               if $verb > 2;
            last;
         }
         _skip_comment ($lref, $h);
         if (! ($$lref =~ s/=>?//) ) {
            print STDERR "\"=>\" erwartet\n";
            last;
         }
         _skip_comment ($lref, $h);
         if (! defined ($v2 = _read_value ($lref, $h))) {
            print STDERR "Hash-Wert erwartet\n" if $verb > 2;
            last;
         }
         $v -> {$k} = $v2;
      }
   }
   elsif ($c eq "\"") {
      $v = "";
      while (1) {
         while ( $$lref =~ s/^([^"]*?)\\(["\\])// ) {
            $v .= $1 . $2;
         }
         if ( $$lref =~ s/^([^"]+)// ) {
            $v .= $1;
         }
         last if $$lref =~ s/^"// ;
         if (!$h || !defined ($$lref = <$h>)) {
            print STDERR "unerwartetes Zeichenketten-Ende: " . $$lref if $verb;
            last;
         }
      }
   }
   else {
      $$lref =~ s/^([^[:space:]{},\[\]=]*)//;
      $v = "$c$1";
   }
   print STDERR "_read_value returns " . (defined $v ? $v : "undef") . "\n" if $verb > 2;
   $v;
} # _read_value


sub _skip_comment {
   my ($lref, $h) = @_;
   while (1) {
      $$lref =~ s/^\s*//;
      $$lref =~ s/^#.*//s;
      last if ($$lref);
      last if !$h || !defined ($$lref = <$h>);
   }
```

```
} # _skip_comment

1;
```

# Herbaer::XMLDataWriter

[Quelltext]

## Anwendung

```
use Herbaer::XMLDataWriter;  # HASH-Daten einfach als XML ausgeben

$data = {
   "kommissare" => [
      { "name" => "Thiel",   "ort"  => "Münster",  },
      { "name" => "Ballauf", "ort"  => "Köln",     },
      ],
   "fahrzeuge" => { "Thiel" => "Fahrrad", "Ballauf" => "Auto", },
   "stand"   => "aktuell",
   "version" => "1",
};
$options = {
   '@kommissare' => ["", "kommissar"],
   '%fahrzeuge'  => ["fahrzeug", "benutzt", '@kommissar'],
};

my $filepath_or_handle = "output.xml";
my $encoding           = "utf-8";
my $namespace          = "xml_tatort";
my $xslt               = "transform.xslt";
my $xmlwriter = Herbaer::XMLDataWriter -> new ($options);
$xmlwriter -> open ($filepath_or_handle, $encoding, $namespace, $xslt);
$xmlwriter -> write ("tatort", {}, $data);
$xmlwriter -> close ();

$xmlwriter -> ignore ('^_');
$xmlwriter -> open ("erweitert.xml");
$xmlwriter -> open_element ("tatort");
$xmlwriter -> comment ("Hoffentlich verletze ich nicht die Rechte der ARD");
$xmlwriter -> write ("kommissare", {}, $data -> {"kommissare"});
$xmlwriter -> write (
   "kommentar", {},
   [ "Münster: genial komisch", "Köln: Schwäche ist menschlich", ]
   );
   $xmlwriter -> meta() -> {"author"} = "Herbär";
$xmlwriter -> write_meta ();
$xmlwriter -> close_element ("tatort");
$xmlwriter -> close ();

print STDERR $xmlwriter -> errormsg ();
```

## Beschreibung

`Herbaer::XMLDataWriter` erlaubt eine einfache Ausgabe einer HASH-Datenstruktur als XML-Dokument. Es ist nicht vorgesehen, eine bestimmte Reihenfolge festzulegen, in der die HASH-Einträge ausgegeben werden.

## Funktionen

Herbaer::XMLDataWriter -> `new` (*`$options`*, *`$enc`*, *`$namespace`*, *`$xslt`*)

*`$options`*

HASH der Ausgabe-Einstellungen.

*`$enc`*

Die Voreinstellung der Kodierung (`encoding`) der Ausgabe. Der Default ist `utf-8`. Der Wert wirkt, wenn beim Aufruf von `open` der Parameter *`$enc`* nicht definiert ist.

*`$namespace`*

Der Vorgabewert für den XML-Namensraum, der mit dem öffnenden Tag des Wurzelelements ausgegeben wird. Der Wert wirkt, wenn beim Aufruf von `open` der Parameter *`$namespace`* nicht definiert ist.

*$xslt*

Der Vorgabewert für den Verweis in der XSLT-Verarbeitungsanweisung. Der Wert wirkt, wenn beim Aufruf von open der Parameter *$xslt* nicht definiert ist. Der Wert "none" bedeutet, dass keine XSLT-Anweisung ausgegeben wird.

Die Funktion new (Konstruktor) ergibt einen neuen Kontext (Objekt) für die XML-Ausgabe. Die Parameter *$enc*, *$namespace* und *$xslt* sind lediglich Vorbelegungen für die entsprechenden Parameter beim Aufruf von open.

$xmlwriter -> open (*$path*, *$enc*, *$namespace*, *$xslt*)

*$path*

Der Dateipfad einer Ausgabedatei, die zu erstellen ist, oder ein geöffnetes Ausgabe-Handle oder "-" für die Ausgabe nach STDOUT. Der Vorgabe-Wert ist "-".

*$enc*

Die Kodierung (encoding) der Ausgabe. Wenn im Ausgabe-Kontext noch kein Vorgabewert definiert ist, wird *$enc* der neue Vorgabe-Wert.

*$namespace*

Der XML-Namensraum, der mit dem öffnenden Tag des Wurzelelements ausgegeben wird. Wenn noch kein Vorgabewert definiert ist (s. new), wird *$namespace* der Vorgabewert.

*$xslt*

Wenn *$xslt* nicht leer und nicht "none" ist, gibt die Funktion open eine XSLT-Verarbeitungsanweisung aus, die auf die URL *$xslt* verweist. Wenn noch kein Vorgabewert definiert ist, wird *$xslt* der neue Vorgabewert.

Alle Parameter sind optional.

Wenn *$path* ein Dateipfad ist, öffnet die Funktion open die Datei zur Ausgabe. Eine existierende Datei oder das Ziel eines symbolischen Verweises werden überschrieben. Das im Pfad angegebene Verzeichnis muss existieren.

Wenn ein Parameterwert leer oder nicht definiert ist, gilt der Vorgabewert des Ausgabe-Kontextes. Wenn im Ausgabe-Kontext kein Vorgabewert definiert ist, wird der übergebene Wert zum Vorgabewert für nachfolgende Aufrufe von open.

open speichert die Systemzeit unter dem Schlüssel date im HASH der Meta-Daten (s. meta). Wenn *$path* ein Dateipfad ist, wird auch *$path* unter dem Schlüssel file gespeichert.

$xmlwriter -> write (*$name*, *$attributes*, *$content*)

*$name*

Ein Name, der die Ausgabe steuert. In der Regel ist *$name* der Name des Elements, das die Funktion write ausgibt.

*$attributes*

HASH-Referenz der Attribute, die zum Element ausgegeben werden. Die Schlüssel sind die Attributnamen, die HASH-Werte die Attributwerte.

*$content*

Der Inhalt des auszugebenden Elements. *$content* kann eine HASH-Referenz, eine ARRAY-Referenz oder ein einfacher skalarer Wert sein. Die Ausgabe hängt wesentlich vom Referenz-Typ von *$content* ab.

Der HASH der Ausgabe-Einstellungen (Konstruktor `new`, Parameter *`$options`*) kann die Ausgabe der Funktion `write` steuern. Der Schlüssel der Ausgabe-Einstellung ist

*`%$name`*, wenn *`$content`* eine HASH-Referenz ist,
*`@$name`*, wenn *`$content`* eine ARRAY-Referenz ist,
*`$$name`*, wenn *`$content`* ein anderer skalarer Wert ist.

Hier beschreibe ich die Ausgabe von `write`, wenn keine Ausgabe-Einstellung unter dem Schlüssel existiert.

Wenn *`$content`* eine HASH-Referenz ist, wird ein Element mit dem Namen *`$name`* und den Attributen *`$attributes`* ausgegeben. Der Inhalt des Elements entsteht durch den Aufruf von `$xmlwriter -> write ($k, undef, $v)` für jedes Schlüssel/Wert-Paar ($k, $v) in *`$content`*.

```
$xmlwriter -> write (
  "kommissar",
  {"alter" => 43,},
  {"name" => "Thiel", "ort" => "Münster"}
);
```

gibt aus:

```
<kommissar alter="43">
  <ort>Münster</ort>
  <name>Thiel</name>
</kommissar>
```

Die Reihenfolge der Kindelemente kann abweichen.

Wenn *`$content`* eine ARRAY-Referenz ist, wird `$xmlwriter -> write ($name, $attributes, $v)` für jeden ARRAY-Eintrag `$v` aufgerufen.

```
$xmlwriter -> write (
   "kommissar",
   {"serie" => "Tatort",},
   ["Thiel", "Ballauf", "Schenk"]
);
```

gibt aus:

```
<kommissar serie="Tatort">Thiel</kommissar>
<kommissar serie="Tatort">Ballauf</kommissar>
<kommissar serie="Tatort">Schenk</kommissar>
```

Wenn *`$content`* ein einfacher Wert ist, wird er als Inhalt eines *`$name`*-Elements ausgegeben. Zeichen, die eine besondere Bedeutung für XML haben, werden durch "character entities" ersetzt, auch in Attributwerten.

```
$xmlwriter -> write (
  "titel",
  {"kommissar" => "Thiel & Börne & \"Alberich\""},
  "Dreimal schwarzer Kater & Mäuschen"
);
```

gibt aus:

```
<titel kommissar="Thiel &amp; Börne &amp; &quot;Alberich&quot;">Dreimal schwarzer Kater &amp; Mäuschen</titel>
```

$xmlwriter -> `close` ()

Schließt die noch geöffneten XML-Elemente und die Ausgabedatei, wenn an `open` oder `new` ein Dateipfad übergeben wurde. Wenn ein geöffnetes Ausgabe-Handle an `open` übergeben wurde, bleibt das Handle geöffnet.

$xmlwriter -> `write_meta` (*`$attributes`*)

Diese Hilfsfunktion gibt "Meta-Daten", die die Funktion `open` anlegt, aus. Sie erspart einen Teil des Schreibaufwandes für `$xmlwriter -> write ("meta", $attributes, $xmlwriter -> meta ())`

$xmlwriter -> `meta` ()

Der Rückgabewert ist die HASH-Referenz der Meta-Daten, die der Funktion `open` anlegt. Vor der Ausgabe der Meta-Daten können weiter Elemente in den HASH eingefügt werden.

```
$xmlwriter -> meta () -> {"author"} = "Herbär";
```

```
$xmlwriter -> write_meta ();
```

gibt etwa aus:

```
<meta>
  <date>2013-02-21T19:18:38</date>
  <file>XMLDataWriter.testout/04_details.xml</file>
  <author>Herbär</author>
</meta>
```

Die Zeitangabe und der Dateipfad sind hier natürlich nur Beispiele.

$xmlwriter -> `ignore` (*`$regex`*)

Der reguläre Ausdruck *`$regex`* passt auf die Namen, die die Funktion `write` bei der Ausgabe ignoriert. Nach `$xmlwriter -> ignore ("^_");` werden z.B. alle HASH-Einträge, deren Schlüssel mit _ beginnen, bei der Ausgabe ignoriert. Tatsächlich wird *`$regex`* mit dem Parameter *`$name`* der Funktion `write` abgeglichen.

Wenn *`$regex`* nicht definiert ist, wird der im Kontext gespeicherte reguläre Ausdruck gelöscht.

```
$xmlwriter -> write ("kommissar", {}, "Thiel");
$xmlwriter -> ignore ("^komm");
$xmlwriter -> write ("kommissar", {}, "Börne");
$xmlwriter -> ignore ();
$xmlwriter -> write ("kommissar", {}, "Schenk");
```

gibt aus:

```
<kommissar>Thiel</kommissar>
<kommissar>Schenk</kommissar>
```

$xmlwriter -> `open_element` (*`$eltname`*, *`$attributes`*)

*`$eltname`*

Der Elementname des auszugebenden öffnenden "Tags".

*`$attributes`*

HASH-Referenz der Attribute das auszugebenden Elements

Diese Funktion gibt das öffnende Tag des Elements mit dem Namen *`$eltname`* und den Attributen *`$attributes`* aus. Der Elementname wird der Liste der offenen Elemente hinzugefügt. Nach dem Aufruf `$xmlwriter -> open_element ($eltname)` sollte später der Aufruf `$xmlwriter -> close_element ($eltname)` folgen. Die Funktion `write` ruft intern `open_element` auf.

```
$xmlwriter -> open_element ("kommissar", {"name" => "Thiel", "ort" => "Münster",});
```

gibt aus:

```
<kommissar ort="Münster" name="Thiel">
```

Die Reihenfolge der Attribute ist nicht festgelegt.

$xmlwriter -> `close_element` (*`$eltname`*)

Diese Funktion schreibt das schließende Tag des Elements mit dem Namen *`$eltname`* und entfernt den ersten Namen in der Liste der Namen der offenen Elemente. Wenn kein Element offen ist oder die beiden Namen nicht übereinstimmen, wird eine Fehlermeldung gespeichert (s. `errormsg`). `close_element` wird intern von `write` aufgerufen.

$xmlwriter -> `commment` (*`$text`*)

Diese Funktion gibt *`$text`* als Kommentar eingerückt in einer neuen Zeile aus und beendet nach dem Kommentar die Zeile. Die Zeichenfolge `--` in *`$text`* wird durch `-*-` ersetzt.

$xmlwriter -> `errormsg` ()

Ergibt einen Text, der in jeder Zeile eine Fehlermeldung enthält. Die möglichen Fehlermeldungen sind:

CANNOT OPEN OUTPUT "*out*": *systemmsg*

Die Datei *out* konnte nicht zur Ausgabe geöffnet werden (open). *systemmsg* ist die System-Fehlermeldung.

NO OPEN ELEMENT in close_element *elt*

close_element ($elt) wurde aufgerufen, aber es ist kein XML-Element offen.

ELEMENT MISMATCH *elt el*

close_element ($elt) wurde mit dem Wert $elt = "*elt*", aufgerufen, aber das offenen Element ist *el*.

OUTPUT NOT OPEN in close_element *elt*

Beim Aufruf von open_element mit dem Parameterwert *elt* ist das Ausgabe-HANDLE nicht geöffnet.

OUTPUT NOT OPEN in write *name*

Beim Aufruf von write mit dem Parameter *$name = "name" elt* ist das Ausgabe-HANDLE nicht geöffnet.

OUTPUT NOT OPEN in comment

Beim Aufruf von comment ist das Ausgabe-HANDLE nicht geöffnet.

UNKNOWN HASH OUTPUT OPTION *name value*

Der Wert *value* der Option zur Ausgabe von HASH-Daten unter dem Schlüssel *name* ist weder eine ARRAY-Referenz noch der Wert IGNORE.

UNKNOWN ARRAY OUTPUT OPTION *name value*

Der Wert *value* der Option zur Ausgabe von ARRAY-Daten unter dem Schlüssel *name* ist weder eine ARRAY-Referenz noch der Wert IGNORE.

UNKNOWN TYPE *name type*

Der Datentyp *type* zur Ausgabe eines einfachen Wertes unter dem Schlüssel *name* in der Funktion write ist nicht bekannt/ungültig.

# Ausgabe-Einstellungen

Die Ausgabe-Einstellungen (new, Parameter *$options*) steuern die Ausgabe der Funktion write (*$name, $attributes, $content*). HASH-Schlüssel der Form [*wort*] (allgemeine Einstellungen) wirken unabängig vom Wert des Parameters *$name* (dem Schlüssel). Andere HASH-Schlüssel wirken nur für einen einzelnen Wert des Parameters *$name*. Diese HASH-Schlüssel sind aus einem Kennzeichen für den Referenz-Typ von *$content* und dem Parameter-Wert *$name* zusammengesetzt. Das Kennzeichen ist

| | |
|---|---|
| % | *$name* ist eine HASH-Referenz |
| @ | *$name* ist eine ARRAY-Referenz |
| $ | sonst |

## Allgemeine Ausgabeeinstellungen

[empty_hash]

Ein logisch wahrer Wert bewirkt, dass zu einem leeren HASH ein XML-Element ausgegeben wird. Auch bei einem logisch wahren Wert erfolgt zu einem leeren HASH keine Ausgabe, wenn in der Einstellung zu dem

jeweiligen Schlüssel ein leerer Wert statt des Namens des Elternelements angegeben ist. Auch für einzelne Schlüssel kann festgelegt werden, dass zu einem leeren HASH ein XML-Element ausgegeben wird.

```
use Herbaer::XMLDataWriter;  # HASH-Daten als XML ausgeben

my $options = {};
my $xmlwriter = new Herbaer::XMLDataWriter ($options);
$xmlwriter -> open ("kommissare.xml");
# ...
my $att     = { "reihe" => "Tatort" };
$xmlwriter -> write ("attribute", $att, {}); # keine Ausgabe
$options -> {"[empty_hash]"} = 1;
$options -> {'%attrib'}      = [""];
$xmlwriter -> write ("attribute", $att, {});
$xmlwriter -> write ("attrib", $att, {});    # keine Ausgabe
$options -> {"[empty_hash]"} = 0;
$xmlwriter -> write ("attribute", $att, {}); # wieder keine Ausgabe
# ...
$xmlwriter -> close ();
```

gibt aus:

```
<attribute reihe="Tatort"></attribute>
```

## Ausgabe unterbinden mit `IGNORE`

Die Zeichenkette "`IGNORE`" als Wert bedeutet, dass keine Ausgabe erfolgt.

```
$xmlwriter -> write ("kommissar", {}, "Kopper");
$options -> {"[empty_hash]"} = 1;
$xmlwriter -> write ("kommissar", {}, "Kopper");
$xmlwriter -> write ("kommissar", {}, ["Odenthal", "Kopper",]);
$xmlwriter -> write ("set",
             {"location" => "Münster"},
             {"kommissar" => "Thiel", "assistent" => "Krusenstern"});
$options -> {'$kommissar'} = undef;
$xmlwriter -> write ("kommissar", {}, "Leitmaier");
$xmlwriter -> write ("kommissar", {}, ["Leitmaier", "Batic",]);
```

gibt aus:

```
<set location="Münster">
  <assistent>Krusenstern</assistent>
</set>
<kommissar>Leitmaier</kommissar>
<kommissar>Leitmaier</kommissar>
<kommissar>Batic</kommissar>
```

`$xmlwriter -> write ("kommissar", {}, "Kopper");` gibt nichts aus, weil die Ausgabe einfacher Werte unter dem Schlüssel "`kommissar`" unterbunden ist. Die Anweisung `$xmlwriter -> write ("kommissar", {}, ["Odenthal", "Kopper",]);` ruft intern `$xmlwriter -> write ("kommissar", {}, "Odenthal");` und `$xmlwriter -> write ("kommissar", {}, "Kopper");` auf und gibt deshalb ebenfalls nichts aus.

```
$options -> {'@kommissar'} = "IGNORE";
$xmlwriter -> write ("kommissar", {}, "Kopper");
$xmlwriter -> write ("kommissar", {}, ["Odenthal", "Kopper",]);
$xmlwriter -> write ("set",
             {"location" => "Köln"},
             {"kommissar" => ["Ballauf", "Schenk",], "assistent" => "Friederike"});
$options -> {'@kommissar'} = undef;
$xmlwriter -> write ("kommissar", {}, ["Ritter", "Stark",]);
```

gibt aus:

```
<kommissar>Kopper</kommissar>
<set location="Köln">
  <assistent>Friederike</assistent>
</set>
<kommissar>Ritter</kommissar>
<kommissar>Stark</kommissar>
```

Das nächste Beispiel unterdrückt die Ausgabe von HASH-Daten.

```
$options -> {'%kommissar'} = "IGNORE";
$xmlwriter -> write ("kommissar", {}, {"name" => "Thiel", "ort" => "Münster",});
$options -> {'%kommissar'} = undef;
$xmlwriter -> write ("kommissar", {}, {"name" => "Lürsen", "ort" => "Bremen",});
```

gibt aus:

```
<kommissar>
  <ort>Bremen</ort>
```

```
  <name>Lürsen</name>
</kommissar>
```

## HASH-Daten

Der Wert zur Steuerung der Ausgabe von HASH-Daten ist eine ARRAY-Referenz von bis zu vier Werten:

```
[$parent, $child, $key, "WRITE_EMPTY"]
```

Wenn der Wert keine ARRAY-Referenz und auch nicht die Zeichenkette `IGNORE` ist, wird ein Fehler gemeldet. Der Wert wird ingnoriert: die Ausgabe erfolgt so, als gäbe es keine Einstellung.

```
my $content = { "name" => "Thiel",   "ort"  => "Münster", };
my $att     = { "sex" => "male" };
$xmlwriter -> write ("kommissar", $att, $content);
$options -> {'%kommissar'} = "krimi";
$xmlwriter -> write ("kommissar", $att, $content);
print STDERR $xmlwriter -> errormsg ();
```

gibt aus:

```
<kommissar sex="male">
  <ort>Münster</ort>
  <name>Thiel</name>
</kommissar>
<kommissar sex="male">
  <ort>Münster</ort>
  <name>Thiel</name>
</kommissar>
```

und schreibt nach STDERR:

```
UNKNOWN HASH OUTPUT OPTION kommissar krimi
```

`$parent` ist der Name des enthaltenden Elements.

```
$options -> {'%kommissar'} = ["ermittler"];
$xmlwriter -> write ("kommissar", $att, $content);
```

gibt aus:

```
<ermittler sex="male">
  <ort>Münster</ort>
  <name>Thiel</name>
</ermittler>
```

Wenn `$parent` nicht definiert oder leer ist, wird kein "umklammerndes" Element ausgegeben. Die Attribute werden in diesem Fall zu den einzelnen Einträgen des HASH ausgegeben.

```
$options -> {'%kommissar'} = [];
$xmlwriter -> write ("kommissar", $att, $content);
$options -> {'%kommissar'} = [''];
$xmlwriter -> write ("kommissar", $att, $content);
```

gibt aus:

```
<ort sex="male">Münster</ort>
<name sex="male">Thiel</name>
<ort sex="male">Münster</ort>
<name sex="male">Thiel</name>
```

`$child` ist der Schlüssel, mit dem die Funktion `write` für jeden HASH-Eintrag aufgerufen wird.

```
$options -> {'%kommissar'} = ["ermittler", "detail"];
$xmlwriter -> write ("kommissar", $att, $content);
$options -> {'%kommissar'} = ['', "detail"];
$xmlwriter -> write ("kommissar", $att, $content);
```

gibt aus:

```
<ermittler sex="male">
  <detail>Münster</detail>
  <detail>Thiel</detail>
</ermittler>
<detail sex="male">Münster</detail>
<detail sex="male">Thiel</detail>
```

Der HASH-Schlüssel kann als Attributwert ausgegeben werden. Dazu ist als `$key` das Zeichen '@', gefolgt vom Namen das Attributs, anzugeben.

```
$options -> {'%kommissar'} = ["ermittler", "detail", '@key'];
```

```
$xmlwriter -> write ("kommissar", $att, $content);
$options -> {'%kommissar'} = ["", "detail", '@key'];
$xmlwriter -> write ("kommissar", $att, $content);
```

gibt aus:

```
<ermittler sex="male">
  <detail key="ort">Münster</detail>
  <detail key="name">Thiel</detail>
</ermittler>
<detail sex="male" key="ort">Münster</detail>
<detail sex="male" key="name">Thiel</detail>
```

Der HASH-Schlüssel wird in den HASH der Attribute eingetragen und nach der Ausgabe wieder gelöscht. Wenn der an `write` übergebene HASH der Attributwerte an die Aufrufe zur Ausgabe der HASH-Einträge übergeben wird, kann der HASH der Attribute geändert werden. Im folgenden Beispiel wird der Schlüssel "`key`" aus dem HASH der Attribute gelöscht.

```
$att = { "key" => "tatort" };
$options -> {'%kommissar'} = undef;
$xmlwriter -> write ("kommissar", $att, $content);
$options -> {'%kommissar'} = ["", "detail", '@key'];
$xmlwriter -> write ("kommissar", $att, $content);
$options -> {'%kommissar'} = undef;
$xmlwriter -> write ("kommissar", $att, $content);
```

gibt aus:

```
<kommissar key="tatort">
  <ort>Münster</ort>
  <name>Thiel</name>
</kommissar>
<detail key="ort">Münster</detail>
<detail key="name">Thiel</detail>
<kommissar>
  <ort>Münster</ort>
  <name>Thiel</name>
</kommissar>
```

Wenn der Wert des HASH-Eintrags selbst eine HASH-Referenz ist, kann der HASH-Schlüssel auch als HASH-Eintrag seines Wertes ausgegeben werden. `$key` ist in diesem Fall der Schlüssel, unter dem der HASH-Schlüssel in den verschachtelten HASH eingetragen wird. `$key` darf in diesem Fall nicht mit dem Zeichen '@' beginnen.

```
my $nestedhash = {
   "Münster" => {
      "kommissar" => "Thiele",
      "assistent" => "Krusenstern",
   },
   "Köln" => {
      "kommissar" => "Ballauf",
      "assistent" => "Friederike",
   }
};
$options -> {'%locations'} = ['', 'location', 'city'];
$xmlwriter -> write ('locations', undef, $nestedhash);
```

gibt aus:

```
<location>
  <city>Köln</city>
  <assistent>Friederike</assistent>
  <kommissar>Ballauf</kommissar>
</location>
<location>
  <city>Münster</city>
  <assistent>Krusenstern</assistent>
  <kommissar>Thiele</kommissar>
</location>
```

Der Eintrag im verschachtelten HASH wird nach der Ausgabe wieder gelöscht. Ein HASH, den ein Eintrag eines auszugebenden HASH referenziert, kann geändert werden. Im folgenden Beispiel wird der Eintrag unter dem Schlüssel "`city`" erst überschrieben und dann gelöscht.

```
$nestedhash = {
   "Köln" => {
      "city"      => "Kalk",
      "kommissar" => "Ballauf",
      "assistent" => "Friederike",
   }
};
$options -> {'%locations'} = ['', 'set', '@city'];
$xmlwriter -> write ('locations', undef, $nestedhash);
$options -> {'%locations'} = ['', 'location', 'city'];
$xmlwriter -> write ('locations', undef, $nestedhash);
$options -> {'%locations'} = ['orte', 'location', '@city'];
```

```
$xmlwriter -> write ('locations', undef, $nestedhash);
```

gibt aus:

```
<set city="Köln">
  <city>Kalk</city>
  <assistent>Friederike</assistent>
  <kommissar>Ballauf</kommissar>
</set>
<location>
  <city>Köln</city>
  <assistent>Friederike</assistent>
  <kommissar>Ballauf</kommissar>
</location>
<orte>
  <location city="Köln">
    <assistent>Friederike</assistent>
    <kommissar>Ballauf</kommissar>
  </location>
</orte>
```

Wenn der Wert des HASH-Eintrags keine HASH-Referenz ist, bleibt ein Wert von `$key`, der nicht mit '@' beginnt, fast (s. unten) wirkungslos.

```
$options -> {'%set'} = ["", "detail", "key"];
$xmlwriter -> write ('set', undef,
  {
     "location" => "Münster",
     "staff"    => { "kommissar" => "Thiele", "medic" => "Börne" },
     "krimis" => [ "Dreimal schwarzer Kater", "Der doppelte Lott", ],
  }
);
```

gibt aus:

```
<detail>Münster</detail>
<detail>
  <medic>Börne</medic>
  <kommissar>Thiele</kommissar>
  <key>staff</key>
</detail>
<detail>Dreimal schwarzer Kater</detail>
<detail>Der doppelte Lott</detail>
```

Wenn `$child` nicht definiert oder leer ist, hängt der zum Aufruf verwendete Schlüssel davon ab, ob `$key` definiert und nicht leer ist. Wenn `$key` definiert und nicht leer ist, gibt `write` den Parameter *`$name`* bei den Aufrufen für die einzelnen HASH-Einträge weiter.

```
$options -> {'%kommissar'} = ["ermittler", undef, '@key'];
$xmlwriter -> write ("kommissar", $att, $content);
```

gibt aus:

```
<ermittler>
  <kommissar key="ort">Münster</kommissar>
  <kommissar key="name">Thiel</kommissar>
</ermittler>
```

Das gilt auch, wenn `$key` nicht mit '@' beginnt und deshalb nicht weiter wirkt:

```
$options -> {'%kommissar'} = ["ermittler", undef, 'key'];
$xmlwriter -> write ("kommissar", $att, $content);
```

gibt aus:

```
<ermittler>
  <kommissar>Münster</kommissar>
  <kommissar>Thiel</kommissar>
</ermittler>
```

In der Regel sind `$child` und `$key` beide definiert und nicht leer oder beide undefiniert.

## Ausgabe bei einem leeren HASH

Zu einem leeren HASH wird in der Regel kein XML-Element ausgegeben, wenn nicht die Einstellung `[empty_hash]` logisch wahr ist. Der Wert `"WRITE_EMPTY"` bewirkt, daß auch zu einem leeren HASH ein XML-Element ausgegeben wird. Der Name des XML-Elements ist `$parent` oder der Schlüssel, falls `$parent` nicht definiert oder leer ist.

```
$att = { "reihe" => "Tatort", "inhalt" => "leer" };
$options -> {'%kommissar'} = ["ermittler", undef, 'key', "WRITE_EMPTY"];
$xmlwriter -> write ("kommissar", $att, {});
```

```
$options -> {'%kommissar'} = ["", undef, 'key', "WRITE_EMPTY"];
$xmlwriter -> write ("kommissar", $att, {});
```

gibt aus:

```
<ermittler reihe="Tatort" inhalt="leer"></ermittler>
<kommissar reihe="Tatort" inhalt="leer"></kommissar>
```

## ARRAY-Daten

Der Wert zur Steuerung der Ausgabe von ARRAY-Daten ist entweder der Wert `IGNORE` oder eine ARRAY-Referenz mit bis zu drei Werten

```
[$parent, $child, "WRITE_EMPTY"]
```

Wenn der Wert keine ARRAY-Referenz und auch nicht die Zeichenkette `IGNORE` ist, wird ein Fehler gemeldet. Der Wert wird ingnoriert: die Ausgabe erfolgt so, als gäbe es keine Einstellung.

`$parent`

> `$parent` ist der Name des umfassenden Elements. Wenn $parent nicht definiert oder leer ist, wird kein umfassendes Element ausgegeben. Die Attribute (*`$attributes`*) werden in diesem Fall zu den einzelen Elementen des ARRAY ausgegeben.

`$child`

> `$child` ist der Schlüssel, der beim Aufruf von `write` zur Ausgabe der einzelnen ARRAY-Elemente übergeben wird.
>
> Wenn `$child` eine ARRAY-Referenz ist, dann werden die Elemente von `$child` nacheinander als Schlüssel zur Ausgabe der einzelnen ARRAY-Elemente an `write` übergeben. Das letzte Element von `$child` wird so oft wie nötig wiederholt.
>
> Wenn `$child` nicht definiert oder leer ist, wird der Wert des Parameters *`$name`* verwendet.

"WRITE_EMPTY"

> Als dritter Eintrag ist die Zeichenkette `"WRITE_EMPTY"` möglich. Normalerweise werden leere ARRAYs nicht ausgegeben. Der Wert `"WRITE_EMPTY"` bewirkt, dass auch zu einem leeren ARRAY ein XML-Element ausgegeben wird. Wenn
> leer oder nicht definiert ist, wird ein leeres ARRAY als XML-Element mit dem Schlüssel als Namen ausgegeben.

Beispiel 1

```
my $kommissare = ["Ballauf", "Schenk"];
$att = { "producer" => "WDR" };
$options -> {'@kommissar'} = [];
$xmlwriter -> write ("kommissar", $att, $kommissare);
```

gibt aus:

```
<kommissar producer="WDR">Ballauf</kommissar>
<kommissar producer="WDR">Schenk</kommissar>
```

Beispiel 2

```
$options -> {'@kommissar'} = [ "ermittler" ];
$xmlwriter -> write ("kommissar", $att, $kommissare);
```

gibt aus:

```
<ermittler producer="WDR">
  <kommissar>Ballauf</kommissar>
  <kommissar>Schenk</kommissar>
</ermittler>
```

Beispiel 3

```
$options -> {'@kommissar'} = [ "", "ermittler" ];
```

```
$xmlwriter -> write ("kommissar", $att, $kommissare);
```

gibt aus:

```
<ermittler producer="WDR">Ballauf</ermittler>
<ermittler producer="WDR">Schenk</ermittler>
```

Beispiel 4

```
$options -> {'@kommissar'} = [ "mordkommission", "ermittler" ];
$xmlwriter -> write ("kommissar", $att, $kommissare);
```

gibt aus:

```
<mordkommission producer="WDR">
  <ermittler>Ballauf</ermittler>
  <ermittler>Schenk</ermittler>
</mordkommission>
```

Beispiel 5

```
$options -> {'@kommissar'} = [ "mordkommission", ["leiter", "ermittler"] ];
$xmlwriter -> write ("kommissar", $att, $kommissare);
```

gibt aus:

```
<mordkommission producer="WDR">
  <leiter>Ballauf</leiter>
  <ermittler>Schenk</ermittler>
</mordkommission>
```

Beispiel 6

```
$options -> {'@kommissar'} = ["", ["leiter", "ermittler"], "WRITE_EMPTY" ];
$xmlwriter -> write ("kommissar", $att, $kommissare);
$xmlwriter -> write ("kommissar", $att, []);
```

gibt aus:

```
<leiter producer="WDR">Ballauf</leiter>
<ermittler producer="WDR">Schenk</ermittler>
<kommissar producer="WDR"></kommissar>
```

Beispiel 7

```
$options -> {'@kommissar'} = "krimi";
$xmlwriter -> write ("kommissar", $att, $kommissare);
print STDERR $xmlwriter -> errormsg ();
```

gibt aus:

```
<kommissar producer="WDR">Ballauf</kommissar>
<kommissar producer="WDR">Schenk</kommissar>
```

und gibt die Meldung `UNKNOWN HASH OUTPUT OPTION kommissar krimi` nach STDERR aus.

## Einfache Werte

Der Wert zur Steuerung der Ausgabe von einfachen Werten ist

entweder der Wert `IGNORE`
oder eine ARRAY-Referenz mit bis zu zwei Werten `[$element, $type]`.
oder eine Zeichenkette `$type` anders als `IGNORE`. Die einfache Zeichenkette `$type` ist gleichbedeutend mit `[undef, $type]`.

`$element` ist der Name des auszugebenden Elements.

```
$att = { "series" => "Tatort" };
$options -> {'$kommissar'} = ["ermittler"];
$xmlwriter -> write ("kommissar", $att, "Börne");
```

gibt aus:

```
<ermittler series="Tatort">Börne</ermittler>
```

Wenn `$element` nicht definiert oder leer ist, wird der Wert des Parameters *`$name`* als Elementname genommen.

```
$att = { "series" => "Tatort" };
$options -> {'$kommissar'} = [];
$xmlwriter -> write ("kommissar", $att, "Thiel");
```

gibt aus:

```
<kommissar series="Tatort">Thiel</kommissar>
```

Die möglichen Werte für `$type` sind:

`empty`

Es wird ein leeres Element ausgegeben, wenn der als Parameter *`$content`* übergebene Wert definiert ist.

```
$options -> {'$kommissar'} = "empty";
$xmlwriter -> write ("kommissar", $att, "Börne");
```

gibt aus:

```
<kommissar series="Tatort"/>
```

`time`

Die Systemzeit (ein ganzzahliger Wert) wird im Format *`YYYY-MM-HH`*`Thh:mm:ss` ausgeben.

```
my $mordzeit = time ();
$xmlwriter -> write ("mordzeit", $att, $mordzeit);
$options -> {'$mordzeit'} = "time";
$xmlwriter -> write ("mordzeit", $att, $mordzeit);
$options -> {'$mordzeit'} = ["crimetime", "time"];
$xmlwriter -> write ("mordzeit", $att, $mordzeit);
```

gibt aus (wenn Ihre Uhr passend gestellt ist):

```
<mordzeit series="Tatort">1362146565</mordzeit>
<mordzeit series="Tatort">2013-03-01T15:02:45</mordzeit>
<crimetime series="Tatort">2013-03-01T15:02:45</crimetime>
```

Andere Werte

Andere Werte führen zu einer Fehlermeldung. Der Wert wird ausgegeben, als wäre `$type` nicht definiert.

```
$options -> {'$mordzeit'} = "Tatzeit";
$xmlwriter -> write ("mordzeit", $att, $mordzeit);
print STDERR $xmlwriter -> errormsg ();
```

gibt aus:

```
<mordzeit series="Tatort">1362147587</mordzeit>
```

und gibt die Meldung `UNKNOWN TYPE mordzeit Tatzeit` an STDERR aus.

# Warnungen

Der an `write` übergebene HASH der Attributwerte (*`$attributes`*) kann in bestimmten Fällen geändert werden, wenn *`$content`* eine HASH-Referenz ist.

Wenn der Parameter *`$content`* der Funktion `write` eine HASH-Referenz ist, kann ein HASH, den ein Eintrag von *`$content`* referenziert, geändert werden.

# Quelltext

[Beschreibung]

```
# Daten als XML ausgeben
# 2013-02-15 Herbert Schiemann <h.schieman@herbaer.de>
# GPL Version 2 oder neuer


# 2020-04-17 open: default STDOUT für out
# 2020-07-29 "Herbaer::*" - Objekte wie HASH
# 2020-12-01 open: Rückgabewert 0 bei Fehler


package Herbaer::XMLDataWriter;


use POSIX qw(strftime);


sub new {
   my ($class, $parm, $encoding, $xmlns, $xslt) = @_;
   my $self = {
      "parm"       => $parm,
      "encoding"   => ($encoding || "utf-8"),
      "xmlns"      => $xmlns,
      "cur_xmlns"  => undef,  # durch "open" für ein Dokument überschrieben
      "xslt"       => $xslt,
      "hout"       => undef,
      "meta"       => {},
      "is_open"    => 0,
      "close_hout" => 0,      # muss hout geschlossen werden?
      "open_elts"  => [],
      "errormsg"   => "",
      "reignore"   => undef,  # RegEx der write-Namen, die ignoriert werden
   };
   return bless ($self, $class);
} # new


# Vorsicht: existierende Datei wird überschrieben
sub open {
   my ($self, $out, $encoding, $xmlns, $xslt) = @_;
   $self -> close () if $self -> {"is_open"};
   $self -> {"encoding"} ||= $encoding;
   $self -> {"xmlns"}    ||= $xmlns;
   $self -> {"xslt"}     ||= $xslt;
   $out      //= "-";
   $encoding ||= $self -> {"encoding"};
   $self -> {"cur_xmlns"} = $xmlns || $self -> {"xmlns"};

   my $meta = $self -> {"meta"};
   $meta -> {"date"} = strftime ("%Y-%m-%dT%H:%M:%S", localtime ());

   my $hout;   # Handle der Ausgabe
   $self -> {"close_hout"} = 0;
   if (ref ($out) eq "GLOB") {
      $hout = $out;
      binmode ($hout, ":raw:encoding($encoding)");
   }
   elsif ($out eq "-") {
      $hout = STDOUT;
      binmode ($hout, ":raw:encoding($encoding)");
   }
   elsif ( open ($hout, ">:encoding($encoding)", $out) ) {
      $meta -> {"file"} = $out;
      $self -> {"close_hout"} = 1;
   }
   else {
      $self -> {"errormsg"} .= "CANNOT OPEN OUTPUT \"$out\": $!\n";
      $hout = undef;
   }
   if ($hout) {
      $self -> {"hout"}    = $hout;
      $self -> {"is_open"} = 1;
      $self -> {"indent"}  = "";
      print $hout "<?xml version=\"1.0\" encoding=\"$encoding\"?>\n";
      $xslt ||= $self -> {"xslt"};
      print $hout "<?xml-stylesheet href=\"$xslt\" type=\"application/xml\"?>\n"
         if $xslt && $xslt ne "none";
      $self -> {"bol"} = 1;   # am Anfang einer Zeile
      $self -> {"open_elts"} = [];
   }
   else {
      return 0;
   }
   $self;
} # open
```

```
sub close {
   $self = shift;
   if ($self -> {"is_open"}) {
      my $hout      = $self -> {"hout"};
      my $open_elts = $self -> {"open_elts"};
      if (@$open_elts) {
         my $indent = $self -> {"indent"};
         my $elt;
         for $elt (@$open_elts) {
            $indent =~ s/^  //;
            print $hout "$indent</$elt>\n";
         }
      }
      close ($hout) if $self -> {"close_hout"};
   }
   $self -> {"hout"}       = undef;
   $self -> {"is_open"}    = 0;
   $self -> {"close_hout"} = 0;
   $self -> {"open_elts"}  = [];
   $self;
} # close


sub attributes {
   my ($self, $att) = @_;
   my $hout = $self -> {"hout"};
   my ($an, $av);
   while ( ($an, $av) = each %$att ) {
      next unless defined $av;
      $av =~ s/&/&amp;/g;
      $av =~ s/</&lt;/g;
      $av =~ s/>/&gt;/g;
      $av =~ s/"/&quot;/g;
      print $hout " $an=\"$av\"";
   }
   $self;
} # attributes


sub open_element {
   my ($self, $elt, $att) = @_;
   if (!$self -> {"is_open"}) {
      $self -> {"errmsg"} .= "OUTPUT NOT OPEN in open_element $elt\n";
   }
   else {
      my $hout   = $self -> {"hout"};
      my $indent = $self -> {"indent"};
      my $open_elts = $self -> {"open_elts"} || [];
      print $hout "\n" unless $self -> {"bol"};
      print $hout ("$indent<$elt");
      if (!@$open_elts && $self -> {"cur_xmlns"}) {
         $att -> {"xmlns"} ||= $self -> {"cur_xmlns"};
      }
      $self -> attributes ($att);
      print $hout ">";
      unshift (@$open_elts, $elt);
      $self -> {"indent"} = "$indent  ";
      $self -> {"bol"}    = 0;
   }
   $self;
} # open_element


sub close_element {
   my ($self, $elt) = @_;
   my $open_elts = $self -> {"open_elts"};
   my $el = shift @$open_elts;
   if (!$el) {
      $self -> {"errormsg"} .= "NO OPEN ELEMENT in close_element $elt\n";
   }
   elsif ($el ne $elt) {
      $self -> {"errormsg"} .= "ELEMENT MISMATCH $elt $el\n";
   }
   if (!$self -> {"is_open"}) {
      $self -> {"errormsg"} .= "OUTPUT NOT OPEN in close_element $elt\n";
   }
   else {
      my $hout = $self -> {"hout"};
      $self -> {"indent"} =~ s/  $//;
      my $indent = $self -> {"indent"};
      print $hout $indent if $self -> {"bol"};
      print $hout "</$elt>\n";
      $self -> {"bol"} = 1;
   }
   $self;
} # close_element
```

```
sub write {
   my ($self, $name, $att, $cont) = @_;
   my $parm   = $self -> {"parm"};
   my $hout   = $self -> {"hout"};
   my $indent = $self -> {"indent"};
   my $reign  = $self -> {"reignore"};
   my $pl;
   $self -> open () if !$self -> {"is_open"};
   if (!defined $cont) {
      return $self;
   }
   elsif ($reign && $name =~ $reign) {
      return $self;
   }
   elsif (!$self -> {"is_open"}) {
      $self -> {"errormsg"} .= "OUTPUT NOT OPEN in write $name\n";
   }
   elsif (ref ($cont) eq "HASH" || ref ($cont) =~ /^Herbaer::/ ) {
      $pl = $parm -> {"\%$name"};
      my $ca;      # Attribute des Kindelements
      my $katt;    # Attributname für den HASH-Schlüssel
      my $pelt;    # umhüllendes Element
      my $kelt;    # Element oder Attribut für die HASH-Schlüssel
      my $celt;    # Element für den HASH-Eintrag
      my $empty;   # Option zur Ausgabe eines leeren HASH
      my ($k, $v); # Schlüssel/Wert-Paar
      if (!$pl) {
         $pelt = $name;
      }
      elsif ($pl eq "IGNORE") {
         return $self;
      }
      elsif (ref ($pl) eq "ARRAY") {
         ($pelt, $celt, $kelt, $empty) = @$pl;
         $empty = 0 unless $empty && $empty eq "WRITE_EMPTY";
      }
      else {
         $self -> {"errormsg"} .= "UNKNOWN HASH OUTPUT OPTION $name $pl\n";
         $pelt = $name;
      }
      if (!%$cont) {
         if ($empty) {
            $pelt ||= $name;
         }
         elsif (! $parm -> {"[empty_hash]"}) {
            return $self;
         }
      }
      if ($pelt) {
         $self -> open_element ($pelt, $att);
         $ca = {};
      }
      else {
         $ca = $att;
      }
      if ($kelt) {
         $celt ||= $name;
         if ($kelt =~ s/^@//) {
            $katt = $kelt;
            while ( ($k, $v) = each %$cont ) {
               $ca -> {$katt} = $k;
               $self -> write ($celt, $ca, $v);
            }
            delete $ca -> {$katt};
         }
         else {
            while ( ($k, $v) = each %$cont ) {
               $v -> {$kelt} = $k
                  if ( ref ($v) eq "HASH" || ref ($v) =~ /^Herbaer::/ );
               $self -> write ($celt, $ca, $v);
               delete $v -> {$kelt}
                  if ( ref ($v) eq "HASH" || ref ($v) =~ /^Herbaer::/ );
            }
         }
      }
      else {
         while ( ($k, $v) = each %$cont ) {
            $self -> write ($celt || $k, $ca, $v);
          }
      }
      $self -> close_element ($pelt) if $pelt;
   }
   elsif ( ref ($cont) eq "ARRAY" ) {
      $pl = $parm -> {"\@$name"};
      my $pelt;
      my $celt;
      my $ca;     # Attribute eines Elements zu einem einzelnen Eintrag
      my $v;
      my $empty;  # Option zur Ausgabe eines leeren ARRAY
      if (!$pl) {}
      elsif (ref ($pl) eq "ARRAY") {
         ($pelt, $celt, $empty) = @$pl;
         $empty = 0 unless $empty && $empty eq "WRITE_EMPTY";
      }
      elsif ( $pl eq "IGNORE" ) {
         return $self;
```

```
      }
      else {
         $self -> {"errormsg"} .= "UNKNOWN ARRAY OUTPUT OPTION $name $pl\n";
      }
      if (!@$cont) {
         return $self unless $empty;
         $pelt ||= $name;
      }
      $celt ||= $name;
      if ($pelt) {
         $self -> open_element ($pelt, $att);
      }
      else {
         $ca = $att;
      }
      if (ref ($celt) eq "ARRAY") {
         my $i = 0;
         my $iend = @$celt - 1;
         for $v (@$cont) {
            $self -> write ($celt -> [$i], $ca, $v);
            ++$i if $i < $iend;
         }
      }
      else {
         for $v (@$cont) {
            $self -> write ($celt, $ca, $v);
         }
      }
      $self -> close_element ($pelt) if $pelt;
   }
   else {
      $pl = $parm -> {"\$$name"};
      my $pelt;
      my $type;
      if (!$pl) {}
      elsif (ref ($pl) eq "ARRAY") {
         ($pelt, $type) = @$pl;
      }
      elsif ( $pl eq "IGNORE" ) {
         return $self;
      }
      else {
         $type = $pl;
      }
      $pelt ||= $name;
      if ($type && $type eq "empty") {
         print $hout $self -> {"indent"};
         print $hout "<$pelt";
         $self -> attributes ($att);
         print $hout "/>\n";
         $self -> {"bol"} = 1;
      }
      else {
         $self -> open_element ($pelt, $att);
         if (!$type) {
            $cont =~ s/&/&amp;/g;
            $cont =~ s/</&lt;/g;
            $cont =~ s/>/&gt;/g;
            print $hout $cont;
         }
         elsif ($type eq "time") {
            print $hout strftime ("%Y-%m-%dT%H:%M:%S", localtime ($cont));
         }
         else {
            $self -> {"errormsg"} .= "UNKNOWN TYPE $name $type\n";
            $cont =~ s/&/&amp;/g;
            $cont =~ s/</&lt;/g;
            $cont =~ s/>/&gt;/g;
            print $hout $cont;
         }
         $self -> close_element ($pelt);
      }
   }
} # write


sub comment {
   my ($self, $text) = @_;
   if (!$self -> {"is_open"}) {
      $self -> {"errmsg"} .= "OUTPUT NOT OPEN in comment\n";
   }
   else {
      my $hout   = $self -> {"hout"};
      my $indent = $self -> {"indent"};
      print $hout "\n" unless $self -> {"bol"};
      $text =~ s/--/-*-/g;
      print $hout ("$indent<!-- $text -->\n");
      $self -> {"bol"} = 1;
   }
   $self;
} # comment


sub write_meta {
   my ($self, $attributes) = @_;
   $self -> write ("meta", $attributes, $self -> {"meta"});
} # write_meta
```

```
sub meta {
   my $self = shift;
   $self -> {"meta"};
} # meta


sub ignore {
   my ($self, $regex) = @_;
   if (defined $regex) {
      $self -> {"reignore"} = qr/$regex/;
   }
   else {
      $self -> {"reignore"} = undef;
   }
   $self;
} # ignore


sub errormsg {
   my $self = shift;
   my $errormsg = $self -> {"errormsg"};
   $self -> {"errormsg"} = "";
   $errormsg;
} # errormsg


1;
```

# tree_sorttlv.xslt

[Quelltext]

## Allgemeines

Knoten der obersten Ebene sortieren

Diese Transformation sortiert die Knoten unterhalb der obersten Knoten in umgekehrter alphabetischer Reihenfolge.

## Namensräume

| Präfix | Namensraum |
|---|---|
| xml | http://www.w3.org/XML/1998/namespace |
| tr | http://herbaer.de/xmlns/20151222/tree/ |
| d | http://herbaer.de/xmlns/20051201/doc |
| xsl | http://www.w3.org/1999/XSL/Transform |

## Ausgabe (output)

| | |
|---|---|
| Method | xml |
| Encoding | utf-8 |

## Muster-Vorlagen (matching templates)

### Muster-Vorlage /tr:tree

Wurzelknoten

### Muster-Vorlage * | @*

Elemente und Attribute werden kopiert

### Muster-Vorlage tr:node

Knoten unterhalb des Dokument-Knotens werden "absteigend" verarbeitet.

# Quelltext

[Beschreibung]

```
<?xml version="1.0" encoding="utf-8"?>
<?xml-stylesheet href="/pool/xslt_ht.xslt" type="application/xml"?>
<!--
  Knoten der zweithöchsten Ebene sortieren
  2017 Herbert Schiemann <h.schiemann@herbaer.de>
  Borkener Str. 167, 46284 Dorsten, Germany
  GPL Version 2 oder neuer
  Jede Gewährleistung ist ausgeschlossen
-->
<xsl:stylesheet
  xmlns:xsl = "http://www.w3.org/1999/XSL/Transform"
  xmlns:d   = "http://herbaer.de/xmlns/20051201/doc"
  xmlns:tr  = "http://herbaer.de/xmlns/20151222/tree/"
  version   = "1.0"
>
<xsl:output method = "xml" encoding = "utf-8"/>


<xsl:template match = "/tr:tree">
  <xsl:copy>
    <xsl:apply-templates select = "tr:node"/>
  </xsl:copy>
</xsl:template>


<xsl:template match = "* | @*">
  <xsl:copy-of select = "."/>
</xsl:template>


<xsl:template match = "tr:node">
  <xsl:copy>
    <xsl:copy-of select = "@*"/>
    <xsl:copy-of select = "* [local-name(.) != 'node']"/>
    <xsl:for-each select = "tr:node">
      <xsl:sort select = "tr:name" data-type = "number" order = "descending"/>
      <xsl:copy-of select = "."/>
    </xsl:for-each>
  </xsl:copy>
</xsl:template>
</xsl:stylesheet>
```

# tree.rng - Baumstruktur von Dokumenten mit Titeln und Verweisen

| | |
|---|---|
| Namespace | http://herbaer.de/xmlns/20151222/tree/ |
| Wurzelelement | tree |
| (foreign_att) | Attribute anderer XML-Namensräume |
| | *Enthalten in:* tree, node, title, ref |
| (anything) | Beliebiger Inhalt |
| | *Enthält:* (anything) (*) |
| | *Enthalten in:* (anything), (foreign_el) |
| (foreign_el) | Elemente anderer XML-Namensräume |
| | *Enthält:* (anything) (*) |
| | *Enthalten in:* tree, node |
| tree | Das Wurzelelement der Baumstruktur |
| | *Enthält:* (foreign_att), (foreign_el), node |
| | *Enthalten in:* Wurzel |

```
<element name="tree">
  <ref name="foreign_att"/>
  <interleave>
    <ref name="foreign_el"/>
    <ref name="el_node"/>
  </interleave>
</element>
```

| | |
|---|---|
| node | Ein Baumknoten |
| | *Enthält:* (foreign_att), (foreign_el), refbase (?), name (?), title, node (*), ref (?) |
| | *Enthalten in:* tree, node |

```
<element name="node">
  <ref name="foreign_att"/>
  <interleave>
    <ref name="foreign_el"/>
    <optional>
        <ref name="el_refbase"/>
    </optional>
    <optional>
        <ref name="el_name"/>
    </optional>
    <ref name="el_title"/>
    <zeroOrMore>
        <ref name="el_node"/>
    </zeroOrMore>
    <optional>
        <ref name="el_ref"/>
    </optional>
  </interleave>
</element>
```

| | |
|---|---|
| refbase | Basis für relative Verweise (`ref`-Elemente) im aktuellen Knoten und Unterknoten. Der Wert ist ein relativer oder absoluter Verzeichnispfad. |
| | *Enthalten in:* node |

```
<element name="refbase">
  <data type="simpleword"/>
</element>
```

| | |
|---|---|
| name | Der Name eines Knotens. Er kann zur Bildung einer Knoten-ID dienen. Er soll unter den "Geschwistern" eindeutig sein. |
| | *Enthalten in:* node |

```
<element name="name">
  <data type="simpleword"/>
</element>
```

title — Der Titel eines Knotens

*Enthält:* Text, (foreign_att)

*Enthalten in:* node

```
<element name="title">
  <ref name="foreign_att"/>
  <text/>
</element>
```

ref — Ein relativer oder absoluter Dateipfad

*Enthält:* Datentyp data

*Enthalten in:* node

```
<element name="ref">
  <ref name="foreign_att"/>
  <data type="data"/>
</element>
```

# xhtml_setlinks.xslt

[Quelltext]

# Namensräume

Die Namensraum-Präfixe, die aus dem erzeugten Dokument ausgeschlossen sind, sind durch einen Stern (*) in der ersten Spalte gekennzeichnet.

| | Präfix | Namensraum |
|---|---|---|
| | xml | http://www.w3.org/XML/1998/namespace |
| | (default) | http://www.w3.org/1999/xhtml |
| * | ht | http://www.w3.org/1999/xhtml |
| * | d | http://herbaer.de/xmlns/20051201/doc |
| | xsl | http://www.w3.org/1999/XSL/Transform |

# Parameter

## Parameter p_href_css

Wert `@href` zu `@rel = "stylesheet"`

Select: '/style/embedded.css'

Der Parameter wird in den folgenden Toplevel-Elementen benutzt:

Muster-Vorlage ht:head

## Parameter p_href_icon

Wert `@href` zu `@rel = "icon"`

Select: '/style/shortcut_icon.png'

Der Parameter wird in den folgenden Toplevel-Elementen benutzt:

Muster-Vorlage ht:head

# Muster-Vorlagen (matching templates)

## Muster-Vorlage /

Wurzel

## Muster-Vorlage ht:body

Der Rumpf wird im Block kopiert.

## Muster-Vorlage ht:head

Im Kopf werden die Verweise auf CSS-Regeln und das Icon eingefügt. Für Handy-Nutzer wird auch ein Element meta[@name="viewport"] eingefügt.

Verwendete globale Parameter oder Variable:

Parameter p_href_css
Parameter p_href_icon

## Muster-Vorlage *

Andere Elemente werden hohl kopiert.

## Muster-Vorlage @* | comment() | processing-instruction()

Attribute, Kommentare und Verarbeitungsanweisungen werden kopiert

# Quelltext

[Beschreibung]

```
<?xml version="1.0" encoding="utf-8"?>
<?xml-stylesheet href="xslt_ht.xslt" type="application/xml"?>
<!--
  Elemente link[@rel = "stylesheet"], link[@rel = "icon"] einfügen
-->
<xsl:stylesheet
  xmlns:xsl = "http://www.w3.org/1999/XSL/Transform"
  xmlns:d   = "http://herbaer.de/xmlns/20051201/doc"
  xmlns:ht  = "http://www.w3.org/1999/xhtml"
  xmlns     = "http://www.w3.org/1999/xhtml"
  exclude-result-prefixes = "d ht"
  version   = "1.0"
>
<xsl:param name = "p_href_css" select = "'/style/embedded.css'"/>


<xsl:param name = "p_href_icon" select = "'/style/shortcut_icon.png'"/>


<xsl:template match = "/">
  <xsl:apply-templates select = "comment() | processing-instruction() | *"/>
</xsl:template>


<xsl:template match = "ht:body">
  <xsl:copy-of select = "."/>
</xsl:template>


<xsl:template match = "ht:head">
  <xsl:copy>
    <xsl:if test = "
      string-length ($p_href_css) &gt; 0
      and not (ht:link[@rel='stylesheet'])
      and not (ht:style)
    ">
      <link href = "{$p_href_css}" rel = "stylesheet"/>
    </xsl:if>
    <xsl:if test = "
      string-length ($p_href_icon) &gt; 0
      and not (ht:link[@rel='icon'])
    ">
      <link href = "{$p_href_icon}" rel = "icon"/>
    </xsl:if>
    <xsl:if test = "not (ht:meta[@name = 'viewport'])">
      <meta name="viewport" content="width=device-width, initial-scale=1"/>
    </xsl:if>
    <xsl:copy-of select = "*"/>
  </xsl:copy>
</xsl:template>


<xsl:template match = "*">
  <xsl:copy>
    <xsl:apply-templates select = "@*|*|text()"/>
  </xsl:copy>
</xsl:template>


<xsl:template match = "@* | comment() | processing-instruction()">
  <xsl:copy-of select = "."/>
</xsl:template>


</xsl:stylesheet>
```

# tree_ht.xslt

[Quelltext]

# Allgemeines

HTML-Darstellung eines Teilbaums

Diese Transformation erzeugt ein XHTML-Dokument. Dessen `xml-stylesheet`-Verarbeitungsanweisung verweist auf `treelist.xslt`.

Dem Teilbaum entspricht ein `ul`-Element mit dem Attribut `class = "tree"`. Dieses Attribut zeigt an, dass das `ul`-Element als „faltbarer" Baum anzuzeigen ist. Das `ul`-Element enthält ein einziges `li`-Element, das dem Wurzelknoten entspricht. Das Attribut `class = "x"` zeigt an, dass der Knoten anfangs „expandiert" angezeigt wird.

Die `li`-Elemente zu einem Dateiknoten enthalten einen Verweis (`a`-Element).

Die `li`-Elemente zu einem Verzeichnisknoten enthalten ein `span`-Element mit einem Verweis auf die Index-Datei sowie ein verschachteltes `ul`-Element, dessen `li`-Kindelemente den Kindknoten entsprechen.

# Namensräume

| Präfix | Namensraum |
|---|---|
| xml | http://www.w3.org/XML/1998/namespace |
| (default) | http://www.w3.org/1999/xhtml |
| xl | http://www.w3.org/1999/xlink |
| ht | http://www.w3.org/1999/xhtml |
| tr | http://herbaer.de/xmlns/20151222/tree/ |
| d | http://herbaer.de/xmlns/20051201/doc |
| xsl | http://www.w3.org/1999/XSL/Transform |

# Ausgabe (output)

| | |
|---|---|
| Method | xml |
| Encoding | utf-8 |

# Parameter

## Parameter p_id

ID des Teilbaums. Sie ist die Verkettung der Namen der Knoten (`tr:name`) vom Wurzelknoten bis zum Teilbaum-Knoten. Die Knotennamen werden durch einen Punkt `.` getrennt.

Select: "

Der Parameter wird in den folgenden Toplevel-Elementen benutzt:

Muster-Vorlage tr:node, search

# Muster-Vorlagen (matching templates)

## Muster-Vorlage /tr:tree

Die Suche nach dem Teilbaum beginnt.

Verwendete Modus:

search

## Muster-Vorlage tr:node, search

### Parameter

pid

ID des Elternelements

Der Teilbaum wird gesucht. Die Verkettung der Namen der übergeordeten Knoten mit dem Punkt als Trennzeichen wird als Parameter `pid` übergeben. Ein Punkt und der Name des Knotens werden angehängt. Falls die Zeichenkette mit dem globalen Parameter `p_id` übereinstimmt, wird ein XHMLT-Dokument erzeugt. Andernfalls werden die untergeordneten Knoten durchsucht.

Verwendete Modus:

item
search

Verwendete globale Parameter oder Variable:

Parameter p_id

## Muster-Vorlage tr:node, item

### Parameter

class

Wert des Attributs @class

refbase

Basispfad für Verweise

HTML-Listeneintrag zu einem Knoten

Verwendete Modus:

item

# Modus

## Modus search

Die folgenden Vorlagen implementieren den Modus search:

Muster-Vorlage tr:node, search

Der Modus search wird in den folgenden Stylesheet-Elementen benutzt:

Muster-Vorlage /tr:tree
Muster-Vorlage tr:node, search

## Modus item

Die folgenden Vorlagen implementieren den Modus item:

Muster-Vorlage tr:node, item

Der Modus item wird in den folgenden Stylesheet-Elementen benutzt:

Muster-Vorlage tr:node, search
Muster-Vorlage tr:node, item

# Quelltext

[Beschreibung]

```
<?xml version="1.0" encoding="utf-8"?>
<?xml-stylesheet href="/pool/xslt_ht.xslt" type="application/xml"?>
<!--
  HTML-Darstellung eines Teilbaums
  2015 Herbert Schiemann <h.schiemann@herbaer.de>
  Borkener Str. 167, 46284 Dorsten, Germany
  GPL Version 2 oder neuer
  Jede Gewährleistung ist ausgeschlossen
-->
<xsl:stylesheet
  xmlns:xsl = "http://www.w3.org/1999/XSL/Transform"
  xmlns:d   = "http://herbaer.de/xmlns/20051201/doc"
  xmlns:tr  = "http://herbaer.de/xmlns/20151222/tree/"
  xmlns:ht  = "http://www.w3.org/1999/xhtml"
  xmlns:xl  = "http://www.w3.org/1999/xlink"
  xmlns     = "http://www.w3.org/1999/xhtml"
  version   = "1.0"
>
<xsl:param name = "p_id" select = "''"/>


<xsl:output method = "xml" encoding = "utf-8"/>


<xsl:template match = "/tr:tree">
  <xsl:apply-templates select = "tr:node" mode = "search">
    <xsl:with-param name = "pid" select = "''"/>
  </xsl:apply-templates>
</xsl:template>


<xsl:template match = "tr:node" mode = "search">
  <xsl:param name = "pid"/>
  <xsl:variable name = "id">
    <xsl:if test = "string-length ($pid)">
      <xsl:value-of select = "concat ($pid, '.')"/>
    </xsl:if>
    <xsl:value-of select = "tr:name"/>
  </xsl:variable>
  <xsl:choose>
    <xsl:when test = "$id = $p_id">
      <xsl:processing-instruction name = "xml-stylesheet">
        <xsl:text>href = "/kal/s/treelist.xslt" type="application/xml"</xsl:text>
      </xsl:processing-instruction>
      <html>
        <xsl:attribute name = "xml:lang">
          <xsl:choose>
            <xsl:when test = "/tr:tree/@xml:lang">
              <xsl:value-of select = "/tr:tree/@xml:lang"/>
            </xsl:when>
            <xsl:otherwise>de</xsl:otherwise>
          </xsl:choose>
        </xsl:attribute>
        <head>
          <title>
            <xsl:value-of select = "tr:title"/>
          </title>
        </head>
        <body>
          <ul class = "tree">
            <xsl:apply-templates select = "." mode = "item">
              <xsl:with-param name = "class"   select = "'x'"/>
              <xsl:with-param name = "refbase" select = "'[none]'"/>
            </xsl:apply-templates>
          </ul>
        </body>
      </html>
    </xsl:when>
    <xsl:otherwise>
      <xsl:apply-templates select = "tr:node" mode = "search">
        <xsl:with-param name = "pid" select = "$id"/>
      </xsl:apply-templates>
    </xsl:otherwise>
  </xsl:choose>
</xsl:template>
```

```
<xsl:template match = "tr:node" mode = "item">
  <xsl:param name = "class"/>
  <xsl:param name = "refbase"/>
  <xsl:variable name = "rb">
    <xsl:choose>
      <xsl:when test = "$refbase = '[none]'"/>
      <xsl:when test = "starts-with (tr:refbase, '/')">
        <xsl:value-of select = "tr:refbase"/>
      </xsl:when>
      <xsl:when test = "
        string-length ($refbase) &gt; 0
        and string-length (tr:refbase) &gt; 0
        and not ($refbase = '/')
      ">
        <xsl:value-of select = "concat ($refbase, '/', tr:refbase)"/>
      </xsl:when>
      <xsl:otherwise>
        <xsl:value-of select = "concat ($refbase, tr:refbase)"/>
      </xsl:otherwise>
    </xsl:choose>
  </xsl:variable>
  <li>
    <xsl:if test = "string-length ($class) &gt; 0">
      <xsl:attribute name = "class">
        <xsl:value-of select = "$class"/>
      </xsl:attribute>
    </xsl:if>
    <xsl:variable name = "href">
      <xsl:choose>
        <xsl:when test = "starts-with (tr:ref, '/')">
          <xsl:value-of select = "tr:ref"/>
        </xsl:when>
        <xsl:when test = "
          string-length ($rb) &gt; 0
          and string-length (tr:ref) &gt; 0
          and not ($rb = '/')
        ">
          <xsl:value-of select = "concat ($rb, '/', tr:ref)"/>
        </xsl:when>
        <xsl:otherwise>
          <xsl:value-of select = "concat ($rb, tr:ref)"/>
        </xsl:otherwise>
      </xsl:choose>
    </xsl:variable>
    <xsl:choose>
      <xsl:when test = "tr:node">
        <span>
          <xsl:choose>
            <xsl:when test = "tr:ref">
              <a href="{$href}">
                <xsl:value-of select = "tr:title"/>
              </a>
            </xsl:when>
            <xsl:otherwise>
              <xsl:value-of select = "tr:title"/>
            </xsl:otherwise>
          </xsl:choose>
        </span>
        <ul>
          <xsl:apply-templates select = "tr:node" mode = "item">
            <xsl:with-param name = "refbase" select = "$rb"/>
          </xsl:apply-templates>
        </ul>
      </xsl:when>
      <xsl:otherwise>
        <a href = "{$href}">
          <xsl:value-of select = "tr:title"/>
        </a>
      </xsl:otherwise>
    </xsl:choose>
  </li>
</xsl:template>


</xsl:stylesheet>
```

# treelist.xslt

[Quelltext]

# Allgemeines

HTML-Listen als Baumansicht

Diese Transformation ist nützlich für XHTML-Dateien mit verschachtelten Listen (`ol` oder `ul`-Elementen) Sie fügt die Dateien `treelist.js` und `treelist.css` hinzu, die die verschachtelten Listen als „faltbare" Baumansicht zeigen.

# Namensräume

Die Namensraum-Präfixe, die aus dem erzeugten Dokument ausgeschlossen sind, sind durch einen Stern (*) in der ersten Spalte gekennzeichnet.

| | **Präfix** | **Namensraum** |
|---|---|---|
| | xml | http://www.w3.org/XML/1998/namespace |
| * | ht | http://www.w3.org/1999/xhtml |
| | (default) | http://www.w3.org/1999/xhtml |
| * | d | http://herbaer.de/xmlns/20051201/doc |
| | xsl | http://www.w3.org/1999/XSL/Transform |

# Muster-Vorlagen (matching templates)

## Muster-Vorlage /

Die Wurzel

## Muster-Vorlage *

Elemente werden absteigend kopiert.

## Muster-Vorlage ht:a [not (@target)]

Verweise führen zu einem neuen Fenster

## Muster-Vorlage @*

Attribute werden direkt kopiert.

## Muster-Vorlage ht:head

Im Kopf werden Stil-Verweise und Skript-Elemente eingefügt

# Quelltext

[Beschreibung]

```
<?xml version="1.0" encoding ="utf-8"?>
<?xml-stylesheet href="/pool/xslt_ht.xslt" type="application/xml"?>
<!--
  HTML-Listen als Baumansicht
  2015 Herbert Schiemann <h.schiemann@herbaer.de>
  Borkener Str. 167, 46284 Dorsten, Germany
  GPL Version 2 oder neuer
  Jede Gewährleistung ist ausgeschlossen
-->
<xsl:stylesheet
  xmlns:xsl = "http://www.w3.org/1999/XSL/Transform"
  xmlns:d   = "http://herbaer.de/xmlns/20051201/doc"
  xmlns     = "http://www.w3.org/1999/xhtml"
  xmlns:ht  = "http://www.w3.org/1999/xhtml"
  exclude-result-prefixes = "d ht"
  version   = "1.0"
  xml:lang  = "de"
>
<xsl:template match = "/">
  <xsl:apply-templates select = "*"/>
</xsl:template>


<xsl:template match = "*">
  <xsl:copy>
    <xsl:apply-templates select = "* | @* | text()"/>
  </xsl:copy>
</xsl:template>


<xsl:template match = "ht:a [not (@target)]">
  <xsl:copy>
    <xsl:attribute name = "target">_blank</xsl:attribute>
    <xsl:apply-templates select = "@* | * | text()"/>
  </xsl:copy>
</xsl:template>


<xsl:template match = "@*">
  <xsl:copy-of select = "."/>
</xsl:template>


<xsl:template match = "ht:head">
  <xsl:if test = "not (ht:link[@rel = 'stylesheet'])">
    <xsl:element name = "link">
      <xsl:attribute name = "rel">stylesheet</xsl:attribute>
      <xsl:attribute name = "href">/kal/s/treelist.css</xsl:attribute>
    </xsl:element>
  </xsl:if>
  <xsl:if test = "not (ht:script)">
    <xsl:element name = "script">
      <xsl:attribute name = "src">/kal/s/treelist.js</xsl:attribute>
    </xsl:element>
  </xsl:if>
  <xsl:if test = "not (ht:link[@rel = 'icon'])">
    <link rel="icon" href = "/style/shortcut_icon.png"/>
  </xsl:if>
  <xsl:if test = "not (ht:meta[@name = 'viewport'])">
    <meta name="viewport" content="width=device-width, initial-scale=1"/>
  </xsl:if>
  <xsl:apply-templates select = "*"/>
</xsl:template>


</xsl:stylesheet>
```

# treelist.js

[Quelltext]

# Funktionalität

Dieses Skript erlaubt verschachtelte Listenstrukturen als einen „faltbaren” Baum darzustellen, dessen Knoten die Surferin durch einen Mausklick ein- und ausblenden kann. Es wird durch die CSS-Regeln `treelist.css` ergänzt.

# Variablen eines Treelist-Objekts

```
var treelist = new Treelist ();
```

Das Objekt `treelist` hat die folgenden Variablen:

`treelist.le`

Ein Element mit diesem Namen ist ein „Container” für Knoten der Baumstruktur. Ein solches Element repräsentiert selbst keinen Knoten, wird aber selbst nach Knoten und Containern durchsucht. Dem Konstruktor kann ein „Container”-Element oder ein „Knotenelement” als Parameter übergeben werden.

Voreingestellt ist der Elementname `ul`.

Ein Containerelement ist nicht erforderlich. Eine Baumansicht kann zum Beispiel nur aus verschachtelten `div`-Elementen als Knoten mit `h2`-Elementen oder `button`-Elementen als „Schaltern” aufgebaut sein. In diesem Fall ist es sinnvoll, wenn die Werte `treelist.le` und `treelist.li` übereinstimmen. Dadurch lassen sich konkrete Baumansichten einfacher erzeugen.

`treelist.li`

Ein Element mit diesem Namen (Knotenelement) repräsentiert einen „Knoten” der Baumansicht. Ein Knoten kann einen „Schalter” (`treelist.se`) enthalten, der die Darstellung des Knotens umschaltet zwischen „expandiert” und „eingefaltet”. Der Schalter ändert nicht die CSS-Regeln zur Darstellung des Knotens direkt, sondern tauscht Wörter (`treelist.kc`, `treelist.kx`) in einem Attribut (`treelist.an`) des Knotenelements aus.

Der Knoten kann weitere untergeordnete Knoten oder Container enthalten. In diesem Fall muss der Schalter vor allen enthaltenen Knoten und Containern stehen, andernfalls wird das Element nicht als Schalter erkannt.

Voreingestellt ist der Elementname `li`.

Bei der Initialisierung einer Baumansicht (einer „Instanz”) werden die Inhalte von Knotenelementen und Containerelementen rekursiv durchlaufen. Knotenelemente werden zusätzlich nach „Schaltern” (`treelist.se`) durchsucht und markiert. Knotenelemente sind in diesem Sinne immer auch Container-Elemente.

`treelist.se`

Ein Kindelement eines Knotens (`treelist.li`) mit diesem Namen repräsentiert einen „Schalter”, der die Darstellung des Knotens umschaltet. Der Ereignisbehandler (`treelist.ev`) dieses Elements tauscht die Wörter `treelist.kc` und `treelist.kx` im Attribut `treelist.an` des Elternelements aus.

Ein Schalter wird nur dann als Schalter erkannt, wenn er vor allen enthaltenen (untergeordneten) Knoten und Containern steht.

Voreingestellt ist der Elementname `span`.

`treelist.kc`

Das Wort `treelist.kc` im Wert des Attributs `treelist.an` kennzeichnet einen Knoten (`treelist.li`) als „eingefaltet”. Die tatsächliche Darstellung bestimmen CSS-Regeln.

Voreingestellt ist das Wort „`c`”.

`treelist.kx`

Das Wort `treelist.kx` im Wert des Attributs `treelist.an` kennzeichnet einen Knoten (`treelist.li`) als „expandiert". Die tatsächliche Darstellung bestimmen CSS-Regeln.

Voreingestellt ist das Wort „`x`".

`treelist.kl`

Knoten (`treelist.li`), in denen kein Schalter (`treelist.se`) erkannt wird, können durch diesen Wert des Attributs `treelist.an` gekennzeichnet werden, falls das Knotenelement anfangs kein Attribut mit diesen Namen hat.

Voreingestellt ist die leere Zeichenkette. In diesem Fall wird das Attribut nicht gesetzt.

`treelist.df`

Wenn diese Zeichenkette definiert und nicht leer ist, bekommen Knoten (`treelist.li`) mit einem Schalter (`treelist.se`), die anfangs kein Attribut mit dem Namen `treelist.an` enthalten, das Attribut `treelist.an` mit diesem Wert (`treelist.df`).

Voreingestellt ist der Wert „`c`". Es entspricht der eingefalteten Darstellung (s. `treelist.kc`)

`treelist.an`

Der Name eines Attributs eines Knotens (`treelist.li`). Bei der Initialisierung wird dieses Attribut gesetzt, falls es nicht bereits existiert. Die Ereignisbehandler (`treelist.ev`) der Schalter (`treelist.se`) tauschen Wörter im Wert dieses Attributs aus.

Voreingestellt ist der Attributname `class`.

`treelist.kn`

Wenn das Attribut `treelist.an` eines Elements mit dem Namen `treelist.li` oder `treelist.le` das Wort `treelist.kn` oder das Wort `treelist.kt` enthält, wird das Element nicht als Knoten oder Container innerhalb eines Knotens, sondern wie anderer Inhalt behandelt.

So ist es möglich, dass innerhalb einer Baumansicht eine Baumansicht verschachtelt ist, die auf eine andere Weise funktioniert.

Voreingestellt ist das Wort „`nofld`".

`treelist.kt`

Wie das Wort `treelist.kn` verhindert dieses Wort im Wert des Attributs `treelist.an`, dass das Element als verschachtelter Container oder Knoten behandelt wird. Das Wort kennzeichnet aber Container-Elemente, bei denen die Initialisierung ansetzt, s. `treelist.mk`.

Voreingestellt ist das Wort „`tree`".

`treelist.ev`

Der Name des Ereignisses, das den „Schalter" drückt. Voreingestellt ist „`click`". Für manche Nutzer könnte „`dblclick`" oder „`mouseover`" sinnvoll sein.

`treelist.mk` (*`el`*)

Die Methode `mk` erzeugt konkrete Baumansichten.

Wenn als Parameter *`el`* ein Element übergeeben wird, so werden Knoten und Container unter den Kindelementen von *`el`* gesucht, die Knoten gekennzeichnet und den Schaltelementen Ereignisbehandler zugeordnet.

Wenn der Parameterwert nicht definiert ist, werden alle Elemente mit dem Namen `treelist.le` und dem Wert `treelist.kt` des Attributs `treelist.an` als Container (der obersten Ebene) behandelt, aber nicht als Knoten.

# Quelltext

[Beschreibung]

```
// -*- coding: utf-8 -*-
/*
  Verschachtelte XHTML-Listen als Baum
  2015-01-28 Herbert Schiemann <h.schiemann@herbaer.de>
*/


// Konstruktor
function Treelist () {
   // Elementname des "Containers", z.B. div, p, ol
   this.le = "ul";

   // Elementname des "Knotens", z.B. p, a, span
   this.li = "li";

   // Name des "Schalters"
   this.se = "span";

   // Attributwort, das einen "eingefalteten" Eintrag (li) kennzeichnet
   this.kc = "c";

   // Attributwort, das einen "expandierten" Eintrag (li) kennzeichnet
   this.kx = "x";

   // Attributwort, das einen "unsensiblen" Blatt-Eintrag (li) kennzeichnet
   // oder leer
   this.kl = "";

   // Default-Anzeige von faltbaren Einträgen (li) ohne class-Attribut
   // "x", "c" oder "" (irgend ein anderes Wort)
   this.df = "c";

   // Name des Attributs, das den Zustand der Knoten kennzeichnet
   this.an = "class";

   // Attributwort, das "Nicht-Knoten" kennzeichnet
   this.kn = "nofld";

   // Attributwort, das einen "Toplevel-Container" kennzeichnet
   this.kt = "tree";

   // Ereignis, auf das der "Schalter" reagiert.
   this.ev = "click";
} // Treelist


Treelist.prototype.mk = function (l) {
   var rc = new RegExp ("\\b" + this.kc + "\\b");
   var rx = new RegExp ("\\b" + this.kx + "\\b");
   var rl = new RegExp ("\\b" + this.kl + "\\b");
   var rn = new RegExp (
      "\\b(?:" + this.kn + "|" + this.kt  + ")\\b"
   );
   var s = this;

   var c;
   var cl;
   var hnd = function (e) {
      c = e.currentTarget;
      while (c && c.nodeType != Node.ELEMENT_NODE) c = c.parentNode;
      if (!c || c.localName == "a") return;
      while (c && c.localName != s.li) c = c.parentNode;
      if (!c) return;
      cl = c.getAttribute (s.an);
      c.setAttribute (
         s.an,
         !cl || cl == ""     ? cl = s.kc            :
         cl.search (rc) >= 0 ? cl.replace (rc, s.kx) :
         cl.search (rx) >= 0 ? cl.replace (rx, s.kc) :
         cl + " " + s.kc
      );
   };

   var es; // erwarte "sensitives" Kindelement des Elements es
   var mt = function () {
      es = null;
      for (
         c = l.firstChild;
         c && c != l;
         c = n
      ) {
         n = null;
         if (c.nodeType == Node.ELEMENT_NODE) {
            // "sensitives Element"
            if (es && c.localName == s.se) {
               c.addEventListener (s.ev, hnd, 1);
               if (s.df) {
                  if (!es.getAttribute (s.an))
                     es.setAttribute (s.an, s.df);
```

```
               }
               es = null;
            }
            // Knoten oder Container
            else if (c.localName == s.li || c.localName == s.le) {
               cl = c.getAttribute (s.an);
               if (!cl || cl.search (rn) < 0) {
                  if (es && s.kl) {
                     // Knoten als nicht faltbar kennzeichnen
                     cl = es.getAttribute (s.an);
                     if (!cl || cl.search (rl) < 0)
                        es.setAttribute (s.an, cl ? cl + " " + s.kl : s.kl);
                  }
                  if (c.localName == s.li)
                     es = c;
//                es = c.localName == s.li ? c : null;
                  n = c.firstChild;
               }
            }
         }
         if (!n) {
            n = c.nextSibling;
            if (!n && es) {
               if (s.kl) {
                  // Knoten als nicht faltbar kennzeichnen
                  cl = es.getAttribute (s.an);
                  if (!cl || cl.search (rl) < 0)
                     es.setAttribute (s.an, cl ? cl + " " + s.kl : s.kl);
               }
               es = null;
            }
         }
         while (!n) {
            c = c.parentNode ;
            n = c.nextSibling ;
         }
      }
   }; // mt

   if (l)
      mt ();
   else {
      var i;
      var nl = document.getElementsByTagName (s.le);
      for (i = 0; i < nl.length; ++i) {
         l = nl [i];
         if (l.getAttribute (s.an) == this.kt) mt ();
      }
   }
}; // Treelist.prototype.mk


onload = function () {
   new Treelist () .mk ();
};
```

# treelist.css

[Quelltext]

## Zweck

Diese Datei definiert Regeln für die Darstellung der unterschiedlichen Zustände von Knoten einer faltbaren Baumansicht. Sie ergänzt die Skriptdatei `treelist.js`. Beide Dateien zusammen werden durch `treelist.xslt` eingebunden.

## Quelltext

[Beschreibung]

```
/*
  CSS-Regeln zur faltbaren Baumansicht
  2016-01-30 Herbert Schiemann <h.schiemann@herbaer.de>
*/


/* Listeneinträge werden nicht zusätzlich markiert */
ul                          { list-style-type: none; }


li                          { margin-top: 0.5em;  }


/* Eingefaltete Knoten */
/* Kindelemente eines eingefalteten Knotens werden nicht angezeigt */
li[class~="c"] > *          { display: none;      }


/* ausgenommen sind die Schalter, hier span-Elemente */
li[class~="c"] > span       {
    display: inline;
    cursor:  pointer;
}


/*
  Vor dem Schalter eines eingefalteten Knotens
  wird ein Dreieck mit der Spitze nach rechts angezeigt
*/
li[class~="c"] > span:before { content: "\25b6\ "; }


/* Expandierte Knoten */
/* Der Zeiger zeigt über dem Schalter an, dass eine Aktion möglich ist. */
li[class~="x"] > span       { cursor: pointer;    }


/*
  Vor dem Schalter eines expandierten Knotens
  wird ein Dreieck mit der Spitze nach unten angezeigt.
*/
li[class~="x"] > span:before { content: "\25bc\ "; }
```

# tree_cmd.xslt

[Quelltext]

## Namensräume

| Präfix | Namensraum |
|---|---|
| xml | http://www.w3.org/XML/1998/namespace |
| (default) | http://www.w3.org/1999/xhtml |
| ht | http://www.w3.org/1999/xhtml |
| tr | http://herbaer.de/xmlns/20151222/tree/ |
| d | http://herbaer.de/xmlns/20051201/doc |
| xsl | http://www.w3.org/1999/XSL/Transform |

## Ausgabe (output)

| | |
|---|---|
| Method | text |
| Encoding | utf-8 |

## Eingebundene Stylesheets

### /pool/txt.xslt - Hilfsvorlagen zur Ausgabe und Verarbeitung von Text

`$txt.break`

## Muster-Vorlagen (matching templates)

### Muster-Vorlage /tr:tree

### Muster-Vorlage tr:node

#### Parameter

pid

Default: "

ID des Elternknotens

refbase

Default: "

Teilpfad zum Elternknoten

Zu Knoten, die Unterknoten enthalten, wird der Befehl ausgegeben, der die passende Index-Datei erzeugt.

# Quelltext

[Beschreibung]

```
<?xml version="1.0" encoding="utf-8"?>
<?xml-stylesheet href="/pool/xslt_ht.xslt" type="application/xml"?>
<!--
  Befehle zum Einfügen der Index-Dateien
  2015 Herbert Schiemann <h.schiemann@herbaer.de>
  Borkener Str. 167, 46284 Dorsten, Germany
  GPL Version 2 oder neuer
  Jede Gewährleistung ist ausgeschlossen
-->
<xsl:stylesheet
  xmlns:xsl = "http://www.w3.org/1999/XSL/Transform"
  xmlns:d   = "http://herbaer.de/xmlns/20051201/doc"
  xmlns:tr  = "http://herbaer.de/xmlns/20151222/tree/"
  xmlns:ht  = "http://www.w3.org/1999/xhtml"
  xmlns     = "http://www.w3.org/1999/xhtml"
  version   = "1.0"
>
<xsl:include href = "/pool/txt.xslt"/>


<xsl:output method = "text" encoding = "utf-8"/>


<xsl:template match = "/tr:tree">
  <xsl:apply-templates select = "tr:node"/>
</xsl:template>


<xsl:template match = "tr:node">
  <xsl:param name = "pid"     select = "''"/>
  <xsl:param name = "refbase" select = "''"/>
  <xsl:if test = "tr:node">
    <xsl:variable name = "id">
      <xsl:choose>
        <xsl:when test = "string-length($pid) &gt; 0 and string-length(tr:name) &gt; 0">
          <xsl:value-of select = "concat ($pid, '.', tr:name)"/>
        </xsl:when>
        <xsl:otherwise>
          <xsl:value-of select = "concat ($pid, tr:name)"/>
        </xsl:otherwise>
      </xsl:choose>
    </xsl:variable>
    <xsl:variable name = "rb">
      <xsl:choose>
        <xsl:when test = "starts-with (tr:refbase, '/')">
          <xsl:value-of select = "tr:refbase"/>
        </xsl:when>
        <xsl:when
            test = "string-length($refbase) &gt; 0 and string-length(tr:refbase) &gt; 0"
            >
          <xsl:value-of select = "concat ($refbase, '/', tr:refbase)"/>
        </xsl:when>
        <xsl:otherwise>
          <xsl:value-of select = "concat ($refbase, tr:refbase)"/>
        </xsl:otherwise>
      </xsl:choose>
    </xsl:variable>
    <xsl:value-of select = "concat ('proc_index ', $id, ' ', $rb, ' ;', $txt.break)"/>
    <xsl:apply-templates select = "tr:node">
      <xsl:with-param name = "pid"     select = "$id"/>
      <xsl:with-param name = "refbase" select = "$rb"/>
    </xsl:apply-templates>
  </xsl:if>
</xsl:template>


</xsl:stylesheet>
```

# tree_files.xslt

[Quelltext]

# Allgemeines

Liste der Dateien in der Baumstruktur

Diese Transformation erstellt eine Liste der relativen URI der Endknoten ohne Unterknoten (Kalenderbildauswahlen). Jeder URI wird in einer Zeile ausgegeben.

# Namensräume

| Präfix | Namensraum |
|---|---|
| xml | http://www.w3.org/XML/1998/namespace |
| (default) | http://www.w3.org/1999/xhtml |
| ht | http://www.w3.org/1999/xhtml |
| tr | http://herbaer.de/xmlns/20151222/tree/ |
| d | http://herbaer.de/xmlns/20051201/doc |
| xsl | http://www.w3.org/1999/XSL/Transform |

# Ausgabe (output)

| | |
|---|---|
| Method | text |
| Encoding | utf-8 |

# Eingebundene Stylesheets

## /pool/txt.xslt - Hilfsvorlagen zur Ausgabe und Verarbeitung von Text

`$txt.break`

# Parameter

## Parameter p_skip

Anzahl der zu überspringenden Ebenen.

Die oberste Ebene ist die Ebene des Eintrags „Kalender” (ID `kal`). Die Ebene darunter enthält die Verzeichniseinträge für einzelne Jahre und die „vorgezogenen” Kalender des aktuellen Jahres. Die „wirklichen” Kalender liegen erst in der dritten Ebene unter einem Jahreseeintrag. Dem entspricht der voreingestellte Wert `2`.

Select: '2'

Der Parameter wird in den folgenden Toplevel-Elementen benutzt:

Muster-Vorlage tr:node

# Muster-Vorlagen (matching templates)

## Muster-Vorlage /tr:tree

## Muster-Vorlage tr:node

### Parameter

skip

Default: $p_skip

Anzahl der zu überspringenden Ebenen

refbase

Default: "

Teilpfad zum Elternknoten

Zu Knoten, die keinen Unterknoten enthalten, wird der Pfad ausgegeben.

Verwendete globale Parameter oder Variable:

Parameter p_skip

# Quelltext

[Beschreibung]

```
<?xml version="1.0" encoding="utf-8"?>
<?xml-stylesheet href="/pool/xslt_ht.xslt" type="application/xml"?>
<!--
  Liste der Dateien in der Baumstruktur
  2015 Herbert Schiemann <h.schiemann@herbaer.de>
  Borkener Str. 167, 46284 Dorsten, Germany
  GPL Version 2 oder neuer
  Jede Gewährleistung ist ausgeschlossen
-->
<xsl:stylesheet
  xmlns:xsl = "http://www.w3.org/1999/XSL/Transform"
  xmlns:d   = "http://herbaer.de/xmlns/20051201/doc"
  xmlns:tr  = "http://herbaer.de/xmlns/20151222/tree/"
  xmlns:ht  = "http://www.w3.org/1999/xhtml"
  xmlns     = "http://www.w3.org/1999/xhtml"
  version   = "1.0"
>
<xsl:param name = "p_skip" select = "'2'"/>


<xsl:include href = "/pool/txt.xslt"/>


<xsl:output method = "text" encoding = "utf-8"/>


<xsl:template match = "/tr:tree">
  <xsl:apply-templates select = "tr:node"/>
</xsl:template>


<xsl:template match = "tr:node">
  <xsl:param name = "skip" select = "$p_skip"/>
  <xsl:param name = "refbase" select = "''"/>
  <xsl:choose>
    <xsl:when test = "tr:node">
      <xsl:variable name = "rb">
        <xsl:choose>
          <xsl:when test = "starts-with (tr:refbase, '/')">
            <xsl:value-of select = "tr:refbase"/>
          </xsl:when>
          <xsl:when
              test = "string-length($refbase) &gt; 0 and string-length(tr:refbase) &gt; 0"
              >
            <xsl:value-of select = "concat ($refbase, '/', tr:refbase)"/>
          </xsl:when>
          <xsl:otherwise>
            <xsl:value-of select = "concat ($refbase, tr:refbase)"/>
          </xsl:otherwise>
        </xsl:choose>
      </xsl:variable>
      <xsl:apply-templates select = "tr:node">
        <xsl:with-param name = "skip"    select = "$skip - 1"/>
        <xsl:with-param name = "refbase" select = "$rb"/>
      </xsl:apply-templates>
    </xsl:when>
    <xsl:when test = "$skip &lt; 1">
      <xsl:variable name = "rf">
        <xsl:choose>
          <xsl:when test = "starts-with (tr:ref, '/')">
            <xsl:value-of select = "tr:ref"/>
          </xsl:when>
          <xsl:when
              test = "string-length($refbase) &gt; 0 and string-length(tr:ref) &gt; 0"
              >
            <xsl:value-of select = "concat ($refbase, '/', tr:ref)"/>
          </xsl:when>
          <xsl:otherwise>
            <xsl:value-of select = "concat ($refbase, tr:ref)"/>
          </xsl:otherwise>
        </xsl:choose>
      </xsl:variable>
      <xsl:value-of select = "concat ($rf, $txt.break)"/>
    </xsl:when>
    <xsl:otherwise/>
  </xsl:choose>
</xsl:template>


</xsl:stylesheet>
```

# kal.xslt

[Quelltext]

# Allgemeines

Bild-Kalender

Diese Transformation erzeugt das HTML-Gerüst für die Kalender sowohl in der Bildschirm- Ansicht als auch in der Druckausgabe. Bildschirmansicht und Druckausgabe nutzen unterschiedliche CSS-Regeln Die Steuerung der Sichtbarkeit der „Blätter” und des Fensters „Einstellung” durch Javascript in der Bildschirmansicht darf die Druckausgabe nicht stören.

Es gibt die folgenden Wörter im Wert der Attribute `class`:

`tiles`

Abschnitt mit den Vorschaubildern („Kacheln”) auf dem „Deckblatt”

`n`

Seitenbereiche für die Bildschirmansicht mit Schaltflächen, die normalerweise nicht sichtbar (transparent) sind.

`hide`

Fenster „Einstellung”: `div`-Element mit der ID `set`

`lft`

`p`-Elemente, deren Inhalt linksbündig statt zentriert ausgegeben werden soll (im Fenster „Einstellung”)

`d`

Bildchen auf dem Deckblatt zu einem Monat.

`img`

Umhüllende `div`- und `a`-Elemente um das Bild eines Monatsblatts

`m`

Bild (`img`-Element) auf einem Monatsblatt

Es gibt die folgenden ID-Werte:

`d00`

Das „Deckblatt”

`bl`

Linker Seitenrandbereich mit dem Dreieck mit der Spitze nach links

`br`

Rechter Seitenrandbereich mit dem Dreieck mit der Spitze nach rechts

`bt`

Oberer Seitenrandbereich mit einigen „Schaltern” und Verweisen.

`ah`

`a`-Element im oberen Seitenrandbereich verweist auf den Anfang des Kalenders.

`aa`

`a`-Element im oberen Seitenrandbereich verweist auf die Startseite der Website.

`as`

`a`-Element im oberen Seitenrandbereich öffnet das Fenster „Einstellung”.

`set`

`div`-Element (Fenster) „Einstellung”

`chk_vm`

Markierungsfeld „Vormonat„

`chk_fm`

Markierungsfeld „Folgemonat„

`chk_kw`

Markierungsfeld „Kalenderwoche„

`chk_mc`

Markierungsfeld „Monatsname in Tabelle„

`b_cl`

Schaltfläche „Schließen„ im Fenster „Einstellung".

`t`*`99`*

`img`-Element des Kalenderblatts für den Monat *`99`*: `01` für Januar, `12` für Dezember.

Weitere ID-Werte und Wörter im Attribut `class` werden aus den Grunddaten übernommen.

# Namensräume

| **Präfix** | **Namensraum** |
|---|---|
| xml | http://www.w3.org/XML/1998/namespace |
| xl | http://www.w3.org/1999/xlink |
| l | http://herbaer.de/xmlns/20141210/localization |
| lt | http://herbaer.de/xmlns/20151212/loctext/ |
| kb | http://herbaer.de/xmlns/20151211/kalenderbilder/ |
| ht | http://www.w3.org/1999/xhtml |
| (default) | http://www.w3.org/1999/xhtml |
| d | http://herbaer.de/xmlns/20051201/doc |
| xsl | http://www.w3.org/1999/XSL/Transform |

# Globale Variable

## Variable g_rootelt

Wurzelelement der Kalenderbilder

Select: /kb:kalenderbilder

Die Variable wird in den folgenden Toplevel-Elementen benutzt:

Benannte Vorlage kalender

# Muster-Vorlagen (matching templates)

## Muster-Vorlage kb:m

Eintrag zu einem Monat

## Muster-Vorlage kb:s

Kennung der Bildergeschichte

## Muster-Vorlage /

## Muster-Vorlage kb:kalenderbilder

Aufgerufene benannte Vorlagen:

kalender

## Muster-Vorlage ht:body/ht:div

### Parameter

l

g_l

In die Abschnitte der Monate werden die Bilder eingefügt

## Muster-Vorlage ht:body/@kb:y

Das Jahr erscheint unter der Überschrift des ersten Abschnitts

## Muster-Vorlage ht:*

### Parameter

cls

g_l

HTML-Elemente werden kopiert

## Muster-Vorlage l:ph

### Parameter

g_l

Platzhalter werden eingesetzt

# Benannte Vorlagen

## Benannte Vorlage kalender

### Parameter

data

Zeichenkette: Titel und Bildauswahl, s. Beschreibung

Erzeugt das HTML-Dokument

Der Parameter `data` ist eine Zeichenkette mit den Daten der Bilder im Format *`TITEL`*`?||((`*`LINK`*`|)?|`*`IMGID`*`||){12}`

*`LINK`* steht für die Kennung der Bildergeschichte ohne das Präfix "`s`", optional gefolgt von einem Fragmentbezeichner.

Der Kopf enthält Verweise auf die Dateien `kal.js`, `kal.css` und `kal.prt`.

Die Vorlage wird aufgerufen in:

Muster-Vorlage kb:kalenderbilder

Aufgerufene benannte Vorlagen:

tiles

Verwendete globale Parameter oder Variable:

Variable g_rootelt

## Benannte Vorlage tiles

### Parameter

l

m

Default: 1

Ein Monatsbild auf der Titelseite

Die Vorlage wird aufgerufen in:

Benannte Vorlage kalender
Benannte Vorlage tiles

Aufgerufene benannte Vorlagen:

tiles

# Quelltext

[Beschreibung]

```
<?xml version="1.0" encoding ="utf-8"?>
<?xml-stylesheet href="/pool/xslt_ht.xslt" type="application/xml"?>
<!--
  Bild-Kalender
  2015 Herbert Schiemann <h.schiemann@herbaer.de>
  Borkener Str. 167, 46284 Dorsten, Germany
  GPL Version 2 oder neuer
  Jede Gewährleistung ist ausgeschlossen
  2017-05-25 g_l lokal, Sprache aus Bildergeschichte
  2017-05-25 lokale Variable home statt g_home
-->
<xsl:stylesheet
  xmlns:xsl = "http://www.w3.org/1999/XSL/Transform"
  xmlns:d   = "http://herbaer.de/xmlns/20051201/doc"
  xmlns     = "http://www.w3.org/1999/xhtml"
  xmlns:ht  = "http://www.w3.org/1999/xhtml"
  xmlns:kb  = "http://herbaer.de/xmlns/20151211/kalenderbilder/"
  xmlns:lt  = "http://herbaer.de/xmlns/20151212/loctext/"
  xmlns:l   = "http://herbaer.de/xmlns/20141210/localization"
  xmlns:xl  = "http://www.w3.org/1999/xlink"
  version   = "1.0"
  xml:lang  = "de"
>


<xsl:variable name = "g_rootelt" select = "/kb:kalenderbilder"/>


<xsl:template match = "kb:m">
  <xsl:apply-templates select = "kb:s"/>
  <xsl:value-of select = "kb:i"/>
  <xsl:text>||</xsl:text>
</xsl:template>


<xsl:template match = "kb:s">
  <xsl:value-of select = "."/>
  <xsl:text>|</xsl:text>
</xsl:template>


<xsl:template match = "/">
  <xsl:apply-templates select = "*"/>
</xsl:template>


<xsl:template match = "kb:kalenderbilder">
  <xsl:call-template name = "kalender">
    <xsl:with-param name = "data">
      <xsl:text>||</xsl:text>
      <xsl:apply-templates select = "kb:m"/>
    </xsl:with-param>
  </xsl:call-template>
</xsl:template>


<xsl:template name = "kalender">
  <xsl:param name = "data"/>
  <xsl:variable name = "pl">
    <xsl:choose>
      <xsl:when test = "string-length(@xml:lang) = 0"/>
      <xsl:when test = "contains (@xml:lang, '-')">
        <xsl:value-of select = "substring-before(@xml:lang, '-')"/>
      </xsl:when>
      <xsl:otherwise>
        <xsl:value-of select = "@xml:lang"/>
      </xsl:otherwise>
    </xsl:choose>
  </xsl:variable>
  <xsl:variable name = "lp">
    <xsl:choose>
      <xsl:when test = "string-length($pl) = 0">
        <!--
Im Falle der Bildergeschichten ist @xml:lang immer definiert.
Hier liegt ein Kalender mit ausgewählten Bilder unter /kal/YYYY/BILDERAUSWAHL.xml vor.
Hier setze ich einen festen relativen Pfad zur Lokalisierungsdatei voraus.
        -->
        <xsl:value-of select = "'../../local/local.xml'"/>
      </xsl:when>
      <xsl:otherwise>
        <xsl:value-of select = "concat ('/local/local.xml.', $pl)"/>
      </xsl:otherwise>
    </xsl:choose>
  </xsl:variable>
  <!--
Der relative Pfad zurück zur Wurzel.
Kalenderbilddatei oder Bildergeschichte?
S. auch Fallunterscheidung in der Vorlage für ht:body/ht:div
  -->
```

```
  <xsl:variable name = "home">
    <xsl:choose>
      <xsl:when test = "string-length($pl) &gt; 0">
      <!--
$pl ist zu Bildergeschichten definiert.
Der Pfad einer Bildergeschichte ist /sID/story.xml
      -->
        <xsl:value-of select = "'..'"/>
      </xsl:when>
      <xsl:otherwise>
      <!--
Zu einer Kalenderbilddatei ist keine Sprache definiert.
Der Pfad ist /kal/YYYY/BILDAUSWAHL.xml
      -->
        <xsl:value-of select = "'../..'"/>
      </xsl:otherwise>
    </xsl:choose>
  </xsl:variable>


  <xsl:variable name = "g_l" select = "document($lp)/l:localization"/>


  <xsl:variable name = "y">
    <xsl:choose>
      <xsl:when test = "$g_rootelt/kb:y">
        <xsl:value-of select = "$g_rootelt/kb:y"/>
        <xsl:text>/a</xsl:text>
      </xsl:when>
      <xsl:otherwise>a</xsl:otherwise>
    </xsl:choose>
  </xsl:variable>
  <xsl:variable name = "body"
    select = "document(concat ('/kal/b/', $y, '.xml'))/ht:body"
  />
  <xsl:variable name = "lang">
    <xsl:variable name = "a" select = "$g_l/@xml:lang"/>
    <xsl:choose>
      <xsl:when test = "string-length ($a) = 0">
        <xsl:text>de</xsl:text>
      </xsl:when>
      <xsl:when test = "contains ($a, '-')">
        <xsl:value-of select = "substring-before ($a, '-')"/>
      </xsl:when>
      <xsl:otherwise>
        <xsl:value-of select = "$a"/>
      </xsl:otherwise>
    </xsl:choose>
  </xsl:variable>
  <xsl:variable name = "title">
    <xsl:choose>
      <xsl:when test = "string-length (substring-before ($data, '||')) &gt; 0">
        <xsl:value-of select = "substring-before ($data, '||')"/>
      </xsl:when>
      <xsl:when test = "not ($g_rootelt/kb:t/lt:v)">
        <xsl:value-of select = "$g_rootelt/kb:t"/>
      </xsl:when>
      <xsl:when test = "$g_rootelt/kb:t/lt:v/lt:t [@l = $lang]">
        <xsl:value-of select = "$g_rootelt/kb:t/lt:v/lt:t [@l = $lang]"/>
      </xsl:when>
      <xsl:otherwise>
        <xsl:value-of select = "$g_rootelt/kb:t/lt:v/lt:t [@l = 'de']"/>
      </xsl:otherwise>
    </xsl:choose>
  </xsl:variable>
  <xsl:variable name = "a" select = "substring-after ($data, '||')"/>
  <html>
    <xsl:copy-of select = "$body/@xml:lang"/>
    <head>
      <meta name="viewport" content="width=device-width, initial-scale=1"/>
      <xsl:element name = "link">
        <xsl:attribute name = "rel">stylesheet</xsl:attribute>
        <xsl:attribute name = "media">screen</xsl:attribute>
        <xsl:attribute name = "href">/kal/s/kal.css</xsl:attribute>
      </xsl:element>
      <xsl:element name = "link">
        <xsl:attribute name = "rel">stylesheet</xsl:attribute>
        <xsl:attribute name = "media">print</xsl:attribute>
        <xsl:attribute name = "href">/kal/s/prt.css</xsl:attribute>
      </xsl:element>
      <link rel="icon" href = "/style/shortcut_icon.png"/>
      <title>
        <xsl:value-of select = "$title"/>
        <xsl:if test = "$g_rootelt/kb:y">
          <xsl:value-of select = "concat (' ', $g_rootelt/kb:y)"/>
        </xsl:if>
      </title>
      <xsl:element name = "script">
        <xsl:attribute name = "src">/kal/s/kal.js</xsl:attribute>
      </xsl:element>
    </head>
    <body>
      <div id="d00">
        <h1><xsl:value-of select = "$title"/></h1>
        <xsl:apply-templates select = "$body/@kb:y"/>
        <div class = "tiles">
          <xsl:call-template name = "tiles">
```

```
            <xsl:with-param name = "l" select = "$a"/>
          </xsl:call-template>
        </div>
      </div>
      <xsl:apply-templates select = "$body/ht:div[2]">
        <xsl:with-param name = "l" select = "$a"/>
        <xsl:with-param name = "g_l" select = "$g_l"/>
      </xsl:apply-templates>
      <div id="bl" class="n">&#x25c0;</div>
      <div id="br" class="n">&#x25b6;</div>
      <div id="bt" class="n">
        <a id = "ah">&#x25c9;</a>
        <a id = "aa" href = "{$home}/index.xhtml" target = "_top">&#x2302;</a>
        <a id = "as">&#x22ee;</a>
        <a href = "{$home}/kal/s/kal_help.xhtml">?</a>
      </div>
      <div id="set" class="hide">
        <h3><l:ph id = "einstell"/></h3>
        <p class = "lft">
          <label>
            <input type = "checkbox" id = "chk_vm"/>
            <l:ph id = "vormonat"/>
          </label>
        </p>
        <p class = "lft">
          <label>
            <input type = "checkbox" id = "chk_fm"/>
            <l:ph id = "folgemonat"/>
          </label>
        </p>
        <p class = "lft">
          <label>
            <input type = "checkbox" id = "chk_kw" checked = "checked"/>
            <l:ph id = "kalenderwoche"/>
          </label>
        </p>
        <p class = "lft">
          <label>
            <input type = "checkbox" id = "chk_mc"/>
            <l:ph id = "monatsname_in_tabelle"/>
          </label>
        </p>
        <p>
          <button id = "b_cl"><l:ph id = "schliessen"/></button>
        </p>
      </div>
    </body>
  </html>
</xsl:template>


<xsl:template name = "tiles">
  <xsl:param name = "l"/>
  <xsl:param name = "m" select = "1"/>
  <xsl:variable name = "mm" select = "substring (100 + $m, 2)"/>
  <xsl:variable name = "i" select = "substring-before ($l, '||')"/>
  <xsl:variable name = "a" select = "substring-after ($l, '||')"/>
  <xsl:variable name = "s">
    <xsl:choose>
      <xsl:when test = "contains ($i, '#')">
        <xsl:value-of select = "substring-before ($i, '#')"/>
      </xsl:when>
      <xsl:when test = "contains ($i, '|')">
        <xsl:value-of select = "substring-before ($i, '|')"/>
      </xsl:when>
      <xsl:otherwise/>
    </xsl:choose>
  </xsl:variable>
  <xsl:variable name = "d">
    <xsl:if test = "string-length ($s) &gt; 0">
      <xsl:value-of select = "concat ('/s', $s, '/')"/>
    </xsl:if>
  </xsl:variable>
  <xsl:variable name = "b">
    <xsl:choose>
      <xsl:when test = "contains ($i, '|')">
        <xsl:value-of select = "substring-after ($i, '|')"/>
      </xsl:when>
      <xsl:otherwise>
        <xsl:value-of select = "$i"/>
      </xsl:otherwise>
    </xsl:choose>
  </xsl:variable>
  <img
      src = "{$d}smallimg/{$b}.jpg"
      class = "d"
      id    = "t{$mm}"
      />
  <xsl:if test = "string-length ($a) &gt; 0">
    <xsl:call-template name = "tiles">
      <xsl:with-param name = "l" select = "$a"/>
      <xsl:with-param name = "m" select = "$m + 1"/>
    </xsl:call-template>
  </xsl:if>
</xsl:template>
```

```
<xsl:template match = "ht:body/ht:div">
  <xsl:param name = "l"/>
  <xsl:param name = "g_l"/>
  <xsl:variable name = "i" select = "substring-before ($l, '||')"/>
  <xsl:variable name = "a" select = "substring-after ($l, '||')"/>
  <xsl:variable name = "s">
    <xsl:choose>
      <xsl:when test = "contains ($i, '#')">
        <xsl:value-of select = "substring-before ($i, '#')"/>
      </xsl:when>
      <xsl:when test = "contains ($i, '|')">
        <xsl:value-of select = "substring-before ($i, '|')"/>
      </xsl:when>
      <xsl:otherwise/>
    </xsl:choose>
  </xsl:variable>
  <xsl:variable name = "d">
    <xsl:if test = "string-length ($s) &gt; 0">
      <xsl:value-of select = "concat ('/s', $s, '/')"/>
    </xsl:if>
  </xsl:variable>
  <xsl:variable name = "h">
    <xsl:if test = "string-length ($s) &gt; 0">
      <xsl:value-of select = "concat ('../../s', $s, '/')"/>
    </xsl:if>
  </xsl:variable>
  <xsl:variable name = "b">
    <xsl:choose>
      <xsl:when test = "contains ($i, '|')">
        <xsl:value-of select = "substring-after ($i, '|')"/>
      </xsl:when>
      <xsl:otherwise>
        <xsl:value-of select = "$i"/>
      </xsl:otherwise>
    </xsl:choose>
  </xsl:variable>
  <xsl:variable name = "f">
    <xsl:if test = "contains ($i, '#')">
      <xsl:value-of
        select = "concat ('#', substring-after (substring-before ($i, '|'), '#'))"
      />
    </xsl:if>
  </xsl:variable>
  <xsl:copy>
    <xsl:copy-of select = "@*"/>
    <xsl:apply-templates select = "ht:h1">
      <xsl:with-param name = "g_l" select = "$g_l"/>
    </xsl:apply-templates>
    <div class = "img">
      <a href = "{$h}story.xml{$f}" target = "_top" class = "img">
      <img
          src = "{$d}images/{$b}.jpg"
          class = "m"
          />
      </a>
    </div>
    <xsl:apply-templates select = "preceding-sibling::*[1]/ht:table">
      <xsl:with-param name = "cls" select = "'v'"/>
      <xsl:with-param name = "g_l" select = "$g_l"/>
    </xsl:apply-templates>
    <xsl:apply-templates select = "ht:* [local-name(.) != 'h1']">
      <xsl:with-param name = "g_l" select = "$g_l"/>
    </xsl:apply-templates>
    <xsl:apply-templates select = "following-sibling::*[1]/ht:table">
      <xsl:with-param name = "cls" select = "'n'"/>
      <xsl:with-param name = "g_l" select = "$g_l"/>
    </xsl:apply-templates>
  </xsl:copy>
  <xsl:if test = "string-length ($a) &gt; 0">
    <xsl:apply-templates select = "following-sibling::ht:div[1]">
      <xsl:with-param name = "l" select = "$a"/>
      <xsl:with-param name = "g_l" select = "$g_l"/>
    </xsl:apply-templates>
  </xsl:if>
</xsl:template>


<xsl:template match = "ht:body/@kb:y">
  <h1><xsl:value-of select = "."/></h1>
</xsl:template>
```

```
<xsl:template match = "ht:*">
  <xsl:param name = "cls"/>
  <xsl:param name = "g_l"/>
  <xsl:copy>
    <xsl:choose>
      <xsl:when test = "string-length ($cls) &gt; 0">
        <xsl:copy-of select = "@*[local-name() != 'class']"/>
        <xsl:attribute name = "class">
          <xsl:if test = "@class">
            <xsl:value-of select = "concat (@class, ' ')"/>
          </xsl:if>
          <xsl:value-of select = "$cls"/>
        </xsl:attribute>
      </xsl:when>
      <xsl:otherwise>
        <xsl:copy-of select = "@*"/>
      </xsl:otherwise>
    </xsl:choose>
    <xsl:apply-templates select = "* | text()">
      <xsl:with-param name = "g_l" select = "$g_l"/>
    </xsl:apply-templates>
  </xsl:copy>
</xsl:template>


<xsl:template match = "l:ph">
  <xsl:param name = "g_l"/>
  <xsl:variable name = "id" select = "@id"/>
  <xsl:value-of select="$g_l/l:t[@id=$id]"/>
</xsl:template>


</xsl:stylesheet>
```

# Auswahl der Bilder

Ich benutze emacs zur Erstellung und Bearbeitung der Bildauswahldateien (XML-Namensraum `http://herbaer.de/xmlns/20151211/kalenderbilder/` `[kalenderbilder.rng]`) Das `elisp`-Skript `kalenderbilder.el` definiert einige hilfreiche Tastenkürzel.

Am Anfang steht das Gerüst einer Bildauswahldatei. Ich sehe mir Bilder der Website im Browser an. Wenn ein Bild für einen Kalender vielleicht geeignet ist, kopiere ich die URL und füge sie in einen Kommentar unter einem oder mehreren Schlagwörtern ein. Die Bildauswahldatei sieht dann zunächst etwa so aus:

```
<?xml version="1.0" encoding="utf-8"?>
<kalenderbilder
  xmlns    = "http://herbaer.de/xmlns/20151211/kalenderbilder/"
  xmlns:lt = "http://herbaer.de/xmlns/20151212/loctext/"
>
  <t><lt:v><lt:t l="de">Schwarz</lt:t></lt:v></t>
  <y>2017</y>
<!--
schwarz
http://kleider/s200701/story.xml.de#s9_1277236_0
http://kleider/s2015w22/story.xml.de#s40_2l5sye5n_0
...

weiß
http://kleider/s200701/story.xml.de#s23_2057588_0
http://kleider/s2011w27/story.xml.de#s24_1dssyfvf_1
...

-->
</kalenderbilder>
```

Natürlich habe ich mehr Schlagwörter als nur `schwarz` und `weiß`, und zu jedem Stichwort viele URL zu Bildern. Am Ende der Liste unter jedem Schlagwort folgt eine Leerzeile.

Ich führe die Einfügeposition in die erste Zeile unter dem Schlagwort (hier `schwarz`). Die Tastenkombination **Control**+**c l** sortiert die Zeilen bis zur nächsten Leerzeile alphabetisch. Die Tastenkombination **Control**+**e l** entfernt Zeilen, die im Block bis zur nächsten Leerzeile mehrfach vorkommen. Die Reihenfolge der ersten Zeilen einer Gruppe gleicher Zeilen bleibt erhalten. Die Tastenkombination **Control**+**c x** mischt die Zeilen bis zur nächsten Leerzeile pseudo-zufällig. Die Tastenkombination **Control**+**c k** wählt die ersten zwölf Bilder für den Kalender aus.

Ich prüfe die ausgewählten Bilder. Zeigt das Bild für Juli vielleicht Schnee oder zeigt das Bild für November hellen Sonnenschein? Ich tausche die Bilder, indem ich zu jedem Bild (`m`-Element) den passenden Wert des Attributs `mm` eintrage, z.B. `01` für Januar, `08` für August. Die Tastenkombination **Control**+**c m** sortiert die Bilder entsprechend.

Die Aufbereitung der Bildauswahl für den Webserver ist einfach: Der Titel wird übersetzt, das Suchmaschinen-Stichwort „`Kalender`" hinzugefügt, und die Transformation `add_sspi.xslt` fügt die richtige XML-Stylesheet-Verarbeitungsanweisung ein.

Die Zuordnung von Schlagwörtern zu Bildern ist zu verbessern. Ich benutze zur Auswahl der Kalenderbilder eine Textdatei (*`DATADIR`*`/tags`) die so aufgebaut ist wie der oben gelistete XML-Kommentar.

# Datei kalenderbilder.el

```
;;; kalenderbilder.el --- http://herbaer.de/xmlns/20130131/kleider
;;; file KLEIDER/web/src/kalender/kalenderbilder.el


;; Copyright 2017 Herbert Schiemann


;; Author: Herbert Schiemann <h.schiemann@herbaer.de>
;; Created: 2017-01-01
;; Modified: 2020-11-29
;; Keywords: kalenderbilder


; 2020-11-29 gui-get-selection statt x-get-selection                              ;


;;; Code:


; Der Cursor muss sich in einem (relativen) URL der folgenden Form befinden:
; s2011w44/story.xml.de#s20_1h4syi9w_2
(defun hbkalenderbilder-url-to-m ()
  "m-Element aus URL"
  (interactive)
  (let
      (
       (rtn nil)
       (pnt (point))
       story
       img
       frgm
       m
       mm
       pt
       (re
        (concat
         "\\(http://\\)?[a-z0-9_./]*?"
         "s\\([a-z0-9_]+\\)/s[a-z.]*\\(#s[0-9]*_\\([a-z0-9]+\\)[^[:space:]]*\\)"
         ))
       )
    (re-search-backward "[^a-z0-9#_.:/]")
    (forward-char)
    (when (looking-at re)
      (setq story (match-string 2))
      (setq frgm  (match-string 3))
      (setq img   (match-string 4))
      (goto-char (point-min))
      (search-forward "</kalenderbilder>")
      (setq pt (- (point) 17))
      (if (re-search-backward "<m\\s-+mm\\s-*=\\s-*\"0*\\([0-9]+\\)\"\\s-*>" (point-min) t)
          (progn
            (setq m (string-to-number (match-string 1)))
            (setq mm (if (< m 12) (format "%02d" (1+ m)) "x"))
            )
        (setq mm "01")
        )
      (goto-char pt)
      (insert "  <m mm = \"" mm "\"><s>" story frgm "</s><i>" img "</i></m>\n")
      (setq rtn t)
      )
    (goto-char pnt)
    rtn
    )
  ) ; hbkalenderbilder-url-to-m


; Kalender aus aufeinander folgenden URL-Zeilen
(defun hbkalenderbilder-make-calendar ()
  "Kalender aus aufeinander folgenden URL-Zeilen"
  (interactive)
  (let
      (
       (c 12)
       )
    (while (> c 0)
      (if (hbkalenderbilder-url-to-m)
          (progn
            (forward-line)
            (setq c (1- c))
            )
        (setq c 0)
        )
      )
    )
  ) ; hbkalenderbilder-make-calendar
```

```
(defun hbkalenderbilder-renum-m ()
  "m-Elemente neu numerieren"
  (interactive)
  (let
      (
       (pnt (point))
       ybeg
       yend
       m
       mm
       )
    (goto-char (point-min))
    (search-forward "<kalenderbilder")
    (setq ybeg (point))
    (search-forward "</kalenderbilder>")
    (setq yend (point))
    (goto-char ybeg)
    (setq m 0)
    (while
        (re-search-forward "<m\\s-*mm\\s-*=\\s-*\"\\([a-z0-9]*\\)\"" yend t)
      (setq m (1+ m))
      (setq mm (format "%02d" m))
      (goto-char (match-beginning 1))
      (delete-region (match-beginning 1) (match-end 1))
      (insert mm)
      )
    (goto-char pnt)
    )
  ) ; hbkalenderbilder-renum-m


(defun hbkalenderbilder-insert-xselection-url ()
  "URL aus X-Selection vor der nächsten Leerzeile oder Kommentarende einfügen"
  (interactive)
  (let
      (
       (url (gui-get-selection))
       )
    (when
        (and
         (string-match-p "^\\(http://\\)?[a-z0-9_./]*?s[a-z0-9_]+/s[a-z.]*#s[0-9]*_[a-z0-9_]+$" url)
         (re-search-forward "\n\\s-*?\n\\|-->" (point-max) t)
         )
      (goto-char (match-beginning 0))
      (or (char-equal (char-after) ?-) (forward-char))
      (insert url "\n")
      (beginning-of-line)
      )
    )
  ) ; hbkalenderbilder-insert-xselection-url


;; Liste sortieren
(defun hb-sort-list (lst comp)
  "Sortiert die Liste lst
comp bestimmt die Sortier-Reihenfolge.
comp wird mit zwei Werten der Liste lst als Parameter aufgerufen.
Das Ergebnis ist t, wenn der erste Wert vor dem zweiten Wert
einsortiert wird.
"
  (if (and lst (cdr lst))
      (let (
            (a lst)
            (b (cdr lst))
            (pre lst)
            )
        (while lst
          (setcdr pre (cdr lst))
          (setq pre lst)
          (setq lst (cdr lst))
          )
        (setq pre (cons nil (hb-sort-list a comp)))
        (setq b (hb-sort-list b comp))
        (setq lst pre)
        (while (and (cdr lst) b)
          (when (funcall comp (car b) (car (cdr lst)))
               (setq a (cdr lst))
               (setcdr lst b)
               (setq b a)
               )
          (setq lst (cdr lst))
          )
        (if b (setcdr lst b))
        (cdr pre)
        )
    lst
    )
  ) ; hb-sort-list
```

```
;; Zeilen sortieren
(defun hb-sort-lines ()
  "Sortiert die Zeilen bis zur nächsten leeren Zeile"
  (interactive)
  (let (
        (pnt    (point))
        (ls     (cons nil nil))
        lend
        ln
        )
    (setq lend ls)
    (beginning-of-line)
    (setq ln (buffer-substring (point) (line-end-position)))
    (while (string-match "[^[:space:]]" ln)
      (setcdr lend (cons ln nil))
      (setq lend (cdr lend))
      (delete-region (point) (1+ (line-end-position)))
      (setq ln (buffer-substring (point) (line-end-position)))
      )
    (setq ls (hb-sort-list (cdr ls) 'string< ))
    (while ls
      (insert (car ls) "\n")
      (setq ls (cdr ls))
      )
    (goto-char pnt)
    )
  ) ; hb-sort-lines


(defun hb-rm-same-lines-in-block ()
  "Entfernt Zeilen, die im Block mehrfach vorkommen"
  (interactive)
  (let (
        (pnt (point))
        pnt1
        ln1
        ln2
        fl
        )
    (beginning-of-line)
    (setq ln1 (buffer-substring (point) (line-end-position)))
    (while (string-match "[^[:space:]]" ln1)
      (setq pnt1 (point))
      (if (= 0 (forward-line))
          (setq ln2 (buffer-substring (point) (line-end-position)))
        (setq ln2 "")
        )
      (while (string-match "[^[:space:]]" ln2)
        (setq fl 0)
        (if (string= ln1 ln2)
            (delete-region (point) (1+ (line-end-position)))
          (setq fl (forward-line))
          )
        (if (= fl 0)
            (setq ln2 (buffer-substring (point) (line-end-position)))
          (setq ln2 "")
          )
        )
      (goto-char pnt1)
      (if (= 0 (forward-line))
          (setq ln1 (buffer-substring (point) (line-end-position)))
        (setq ln1 "")
        )
      )
    (goto-char pnt)
    )
  )
 ; hb-rm-same-lines-in-block
```

```
(defun hbkalenderbilder-sort-m ()
  "Sortiert die m-Elemente nach dem Attribut mm"
  (interactive)
  (let (
        (ls     (cons nil nil))
        lend
        m
        )
    (setq lend ls)
    (goto-char (point-min))
    (search-forward "<kalenderbilder" (point-max) t)
    (while (re-search-forward "\\s-*\\(<m.+?</m>\\)\\s-*" (point-max) t)
      (setq m (match-string 1))
      (goto-char (match-beginning 0))
      (delete-region (match-beginning 0) (match-end 0))
      (setcdr lend
              (cons
               (cons
                m
                (if (string-match "<m\\s-+mm\\s-*=\\s-*\"\\([0-9]+\\)\"" m)
                    (string-to-number (match-string 1 m))
                  99
                  )
                )
               nil
               )
              )
      (setq lend (cdr lend))
      )
    (setq ls
          (hb-sort-list (cdr ls)
                        (lambda (a b) (< (cdr a) (cdr b)))
                        )
          )
    (goto-char (point-min))
    (search-forward "</kalenderbilder>" (point-max) t)
    (goto-char (match-beginning 0))
    (insert "\n")
    (while ls
      (insert "  " (car (car ls)) "\n")
      (setq ls (cdr ls))
      )
    )
  ) ; hbkalenderbilder-sort-m


(defun hbkalenderbilder-random-line-to-top ()
  "Vertauscht die aktuelle Zeile mit einer pseudo-zufällig gewählten Folgezeile"
  (interactive)
  (let
      (
       (lb (line-beginning-position))
       le
       (nl 0)
       ln1
       ln2
       )
    (beginning-of-line)
    (while (not (looking-at "\\s*?\n"))
      (forward-line)
      (setq nl (1+ nl))
      )
    (goto-char lb)
    (and (> nl 0) (setq nl (random nl)))
    (if (= nl 0)
        (forward-line)
      (setq le (1+ (line-end-position)))
      (setq ln1 (buffer-substring (point) le))
      (delete-region lb le)
      (setq nl (1- nl))
      (while (> nl 0)
        (forward-line)
        (setq nl (1- nl))
        )
      (setq le (1+ (line-end-position)))
      (setq ln2 (buffer-substring (point) le))
      (delete-region (point) le)
      (insert ln1)
      (goto-char lb)
      (insert ln2)
      )
    )
  ) ; hbkalenderbilder-random-line-to-top
```

```
(defun hbkalenderbilder-shuffle-lines ()
  "Mischt Zeilen pseudo-zufällig"
  (interactive)
  (random t)
  (beginning-of-line)
  (let (
        (pnt (point))
       )
    (while (not (looking-at "\\s*?\n"))
      (hbkalenderbilder-random-line-to-top)
      )
    (goto-char pnt)
    )
  ) ; hbkalenderbilder-shuffle-lines


(let ((map (make-sparse-keymap)))
  (define-key map "\C-ce"    'hb-rm-same-lines-in-block)
  (define-key map "\C-ci"    'hbkalenderbilder-insert-xselection-url)
  (define-key map "\C-ck"    'hbkalenderbilder-make-calendar)
  (define-key map "\C-cl"    'hb-sort-lines)
  (define-key map "\C-cm"    'hbkalenderbilder-sort-m)
  (define-key map "\C-cr"    'hbkalenderbilder-renum-m)
  (define-key map "\C-cs"    'hbkalenderbilder-url-to-m)
  (define-key map "\C-ct"    'hbkalenderbilder-random-line-to-top)
  (define-key map "\C-cx"    'hbkalenderbilder-shuffle-lines)
  (set-keymap-parent map nxml-mode-map)
  (use-local-map map)
  )


;;; kalenderbilder.el ends here
;;; end of file KLEIDER/web/src/kalender/kalenderbilder.el
```

# add_sspi.xslt

[Quelltext]

# Allgemeines

Stylesheet-Verarbeitungsanweisung einfügen

Diese Transformation fügt eine `xml-stylesheet` - Verarbeitungsanweisung ein, wenn es eine solche nicht gibt.

# Namensräume

| Präfix | Namensraum |
|---|---|
| xml | http://www.w3.org/XML/1998/namespace |
| d | http://herbaer.de/xmlns/20051201/doc |
| xsl | http://www.w3.org/1999/XSL/Transform |

# Parameter

## Parameter p_ss

Verweis auf die Transformation

Select: 's/kal.xslt'

Der Parameter wird in den folgenden Toplevel-Elementen benutzt:

Muster-Vorlage /

# Muster-Vorlagen (matching templates)

## Muster-Vorlage /

Verwendete globale Parameter oder Variable:

Parameter p_ss

## Muster-Vorlage *

Elemente werden kopiert.

## Muster-Vorlage processing-instruction()

Verarbeitungsanweisungen werden kopiert.

# Quelltext

[Beschreibung]

```
<?xml version="1.0" encoding ="utf-8"?>
<?xml-stylesheet href="/pool/xslt_ht.xslt" type="application/xml"?>
<!--
  Stylesheet-Verarbeitungsanweisung einfügen
  2017 Herbert Schiemann <h.schiemann@herbaer.de>
  Borkener Str. 167, 46284 Dorsten, Germany
  GPL Version 2 oder neuer
  Jede Gewährleistung ist ausgeschlossen
-->
<xsl:stylesheet
  xmlns:xsl = "http://www.w3.org/1999/XSL/Transform"
  xmlns:d   = "http://herbaer.de/xmlns/20051201/doc"
  version   = "1.0"
  xml:lang  = "de"
>
<xsl:param name = "p_ss" select = "'s/kal.xslt'"/>


<xsl:template match = "/">
  <xsl:if test = "not (processing-instruction() [name() = 'xml-stylesheet'])">
    <xsl:processing-instruction name = "xml-stylesheet">
      <xsl:text>href="</xsl:text>
      <xsl:value-of select = "$p_ss"/>
      <xsl:text>" type="application/xml"</xsl:text>
    </xsl:processing-instruction>
  </xsl:if>
  <xsl:apply-templates select = "* | processing-instruction()"/>
</xsl:template>


<xsl:template match = "*">
  <xsl:copy-of select = "."/>
</xsl:template>


<xsl:template match = "processing-instruction()">
  <xsl:copy-of select = "."/>
</xsl:template>


</xsl:stylesheet>
```

# Verschiedene Sprachen

Ich biete die Kalender in verschiedenen Sprachen an.

In den Kalenderdateien (Bildauswahldateien) sind nur der Titel und das Suchmaschinen-Stichwort „`Kalender`" mit Jahreszahl in verschiedene Sprachen zu übersetzen. Jede einzelne Bildauswahldatei enthält den Titel und das Suchmaschinen-Stichwort in allen angebotenen Sprachen.

Für den Titel in verschiedenen Sprachen nutze ich den XML-Namensraum `http://herbaer.de/xmlns/20151212/loctext/ [loctext.rng]`. In die Quelldateien schreibe ich nur den deutschen Titel und vielleicht ein paar fremdsprachige Titel, sofern meine Sprachkenntnis reicht. Die Übersetzung in andere Sprachen läuft folgendermaßen: Die Transformation `lt_in.xslt` gibt die zu übersetzenden Texte an das Programm `trans.pl`. Dieses gibt die Übersetzungen in einer XML-Datei aus. Die Transformation `lt_merge.xslt` fügt die neuen Übersetzungen in das Dokument ein. Zu jeder Übersetzung gibt es eine Kennung, die die beteiligten Übersetzungsprogramm-Module bezeichnet (Übersetzerkennung). Die Kennung kann nützlich sein, wenn ich einige Titel in bestimmte Sprachen neu übersetzen will. Sonst hat sie keine große Bedeutung. Die Transformation `lt_chktrname.xslt` entfernt oder ersetzt die Übersetzerkennung. Ich ersetze die Kennung durch „Google Translate", falls der Webdienst Google Translate [https://cloud.google.com/translate/] die Übersetzung geliefert hat, und entferne sie sonst. Ich hoffe, damit den Nutzungbedingungen von Google Translate zu genügen.

Nachdem der Titel übersetzt ist, fügt die Transformation `kal_addkeywords.xslt` das Suchmaschinen-Stichwort „`Kalender`" mit der Jahreszahl in den Sprachen des Titels ein. Die Übersetzungen kommen aus den zentralen Lokalisierungsdateien `/local/local.xml.`*`LANG`*

Woran erkennt der Browser, in welcher Sprache der Titel anzuzeigen ist? Die darstellende Transformation `kal.xslt` benötigt die Lokalisierungsdatei, die auch die Sprache angibt. Zu einer Bildergeschichte ist eine Sprache definiert, zu einer Kalenderbildauswahl nicht. So kann ich den relativen Pfad zur Wurzel in den beiden Fällen bestimmen und im Falle einer Bildergeschichte die Sprache der Bildergeschichte wählen. Die Anzeige in chinesischer Sprache bedeutet nicht, dass auch die Feiertage in China angezeigt werden. Dazu müssen die Kalendergrunddaten für China gewählt werden.

Die Hilfe zur Darstellung ist eine XHTML-Datei, die wie andere XHTML-Dateien in verschiedene Sprachen übersetzt wird.

Die Kalender-Grunddaten enthalten Platzhalter für die Monatsnamen und sind so in gewisser Weise unabhängig von der Sprache.

# loctext.rng - Text in verschiedenen Sprachen

Dieser Namensraum wird in zweifacher Funktion benutzt: für Texte in verschiedenen Sprachversionen in einem „fremden” XML-Dokument (Element `v`) und für ein eigenständiges Dokument (Wurzelelement `g`) mit mehreren Sprachversionen von Texten, die in ein fremdes Dokument eingefügt werden. S. `lt_in.xslt`, `trans.pl` und `lt_merge.xslt`

| | |
|---|---|
| Namespace | http://herbaer.de/xmlns/20151212/loctext/ |
| (foreign_att) | Attribute anderer XML-Namensräume |
| | *Enthalten in:* g, v, d, trname, text, t |
| (anything) | Beliebiger Inhalt |
| | *Enthält:* (anything) (*) |
| | *Enthalten in:* (anything), (foreign_el) |
| (foreign_el) | Elemente anderer XML-Namensräume |
| | *Enthält:* (anything) (*) |
| | *Enthalten in:* g, v, d |
| g | Das Wurzelelement eines eigenständigen XML-Dokuments. Ein solches Dokument wird vom Programm `trans.pl` erzeugt. Die Transformation `lt_merge.xslt` fügt die Sprachvarianten in ein anderes XML-Dokument ein. |
| | *Enthält:* (foreign_att), (foreign_el), v (+) |

```
<element name="g">
  <ref name="foreign_att"/>
  <interleave>
    <ref name="foreign_el"/>
    <oneOrMore>
        <ref name="el_v"/>
    </oneOrMore>
  </interleave>
</element>
```

| | |
|---|---|
| v | Umfassendes Element für Varianten eines Textes in verschiedenen Sprachen in einem XML-Dokument eines „fremden” Namensraums. |
| | Die beiden Inhaltsmodelle haben dieselbe Bedeutung. Die `t`-Kindelemente sind für die Einbettung in ein „fremdes” XML-Dokument gedacht. Die `d`-Kindelemente erfordern mehr Tags. Sie sind zur einfachen Ausgabe durch ein Programm gedacht. |
| | *Enthält:* (foreign_att), (foreign_el), t (+), d (+) |
| | *Enthalten in:* g |

```
<element name="v">
  <ref name="foreign_att"/>
  <interleave>
    <ref name="foreign_el"/>
    <choice>
        <oneOrMore>
          <ref name="el_t"/>
        </oneOrMore>
        <oneOrMore>
          <ref name="el_d"/>
        </oneOrMore>
    </choice>
  </interleave>
</element>
```

| | |
|---|---|
| d | Ein Text in einer bestimmten Sprache |
| | *Enthält:* (foreign_att), @l, (foreign_el), trname (?), text |
| | *Enthalten in:* v |

```
<element name="d">
  <ref name="foreign_att"/>
  <ref name="att_l"/>
  <interleave>
    <ref name="foreign_el"/>
    <optional>
        <ref name="el_trname"/>
    </optional>
    <ref name="el_text"/>
  </interleave>
</element>
```

trname — Die Kennung des Übersetzers (Übersetzungsprogramms).

*Enthält:* Datentyp word

*Enthalten in:* d

```
<element name="trname">
  <ref name="foreign_att"/>
  <data type="word"/>
</element>
```

text — Die (übersetzte) Text.

*Enthält:* Datentyp text

*Enthalten in:* d

```
<element name="text">
  <ref name="foreign_att"/>
  <data type="text"/>
</element>
```

t — Ein Text in einer bestimmten Sprache

*Enthält:* Text, (foreign_att), @l, @tr (?)

*Enthalten in:* v

```
<element name="t">
  <ref name="foreign_att"/>
  <ref name="att_l"/>
  <optional>
    <ref name="att_tr"/>
  </optional>
  <text/>
</element>
```

@tr — Die Kennung des Übersetzers (Übersetzungsprogramms).

*Enthalten in:* t

@l — Die Kennung der Sprache eines Textes.

*Enthalten in:* d, t

# lt_in.xslt

[Quelltext]

## Allgemeines

Zu übersetzende Texte extrahieren

Diese Transformation erstellt eine Textdatei mit den zu übersetzenden Texten zur Weiterverarbeitung. Das Format der Ausgabe ist als Eingabeformat des Programms `trans.pl` beschrieben.

## Namensräume

| Präfix | Namensraum |
|---|---|
| xml | http://www.w3.org/XML/1998/namespace |
| lt | http://herbaer.de/xmlns/20151212/loctext/ |
| d | http://herbaer.de/xmlns/20051201/doc |
| xsl | http://www.w3.org/1999/XSL/Transform |

## Ausgabe (output)

| | |
|---|---|
| Method | text |
| Encoding | utf-8 |

## Eingebundene Stylesheets

### /pool/txt.xslt - Hilfsvorlagen zur Ausgabe und Verarbeitung von Text

Zeilenende: $txt.break

## Parameter

### Parameter p_default_lang

Die Sprache eines Textes, wenn eine weitere Angabe fehlt.

Select: 'de'

Der Parameter wird in den folgenden Toplevel-Elementen benutzt:

Variable g_lang

## Globale Variable

### Variable g_lang

Die Sprache, wenn das Attribut lt:@l fehlt

Verwendete globale Parameter oder Variable:

Parameter p_default_lang

Die Variable wird in den folgenden Toplevel-Elementen benutzt:

Muster-Vorlage lt:t

# Muster-Vorlagen (matching templates)

## Muster-Vorlage /

Wurzel des XLIFF-Dokuments

## Muster-Vorlage *

XML-Elemente werden nach lt-Elementen durchsucht.

## Muster-Vorlage lt:v

Ein Text in verschiedenen Sprachen

## Muster-Vorlage lt:t

Eine Sprachvariante

Verwendete globale Parameter oder Variable:

Variable g_lang

# Quelltext

[Beschreibung]

```
<?xml version="1.0" encoding="utf-8"?>
<?xml-stylesheet href="/pool/xslt_ht.xslt" type="application/xml"?>
<!--
  Zu übersetzende Texte extrahieren
  2015 Herbert Schiemann <h.schiemann@herbaer.de>
  Borkener Str. 167, 46284 Dorsten, Germany
  GPL Version 2 oder neuer
  Jede Gewährleistung ist ausgeschlossen
-->
<xsl:stylesheet
  xmlns:xsl = "http://www.w3.org/1999/XSL/Transform"
  xmlns:d   = "http://herbaer.de/xmlns/20051201/doc"
  xmlns:lt = "http://herbaer.de/xmlns/20151212/loctext/"
  version   = "1.0"
>
<xsl:param name = "p_default_lang" select  = "'de'"/>


<xsl:include href = "/pool/txt.xslt"/>


<xsl:variable name = "g_lang">
  <xsl:choose>
    <xsl:when test = "/*/@xml:lang">
      <xsl:value-of select = "/*/@xml:lang"/>
    </xsl:when>
    <xsl:otherwise>
      <xsl:value-of select = "$p_default_lang"/>
    </xsl:otherwise>
  </xsl:choose>
</xsl:variable>


<xsl:output method = "text" encoding = "utf-8"/>


<xsl:template match = "/">
  <xsl:apply-templates select = "*"/>
</xsl:template>


<xsl:template match = "*">
  <xsl:apply-templates select = "*"/>
</xsl:template>


<xsl:template match = "lt:v">
  <xsl:choose>
    <xsl:when test = "@id">
      <xsl:value-of select = "concat ('ID ', @id)"/>
    </xsl:when>
    <xsl:otherwise>
      <xsl:value-of select = "concat ('POS ', count (preceding::lt:v) + 1)"/>
    </xsl:otherwise>
  </xsl:choose>
  <xsl:value-of select = "$txt.break"/>
  <xsl:apply-templates select = "lt:t"/>
</xsl:template>


<xsl:template match = "lt:t">
  <xsl:text>LANG </xsl:text>
  <xsl:choose>
    <xsl:when test = "@l">
      <xsl:value-of select = "@l"/>
    </xsl:when>
    <xsl:otherwise>
      <xsl:value-of select = "$g_lang"/>
    </xsl:otherwise>
  </xsl:choose>
  <xsl:value-of select = "concat ($txt.break, normalize-space (.), $txt.break)"/>
</xsl:template>


</xsl:stylesheet>
```

# trans.pl

[Quelltext]

## Übersicht

`trans.pl --help | --version`

`trans.pl [ --verbose … | --no_verbose ]`
`[ --out` *`OUT`* `] [ --trname` *`TRNAME`* `] [ --srclang` *`SRCLANG`* `] [ --tgtlang` *`TGTLANG`* `] [ --retrans | --no_retrans ]`

## Beschreibung

Dieses Programm dient zur Übersetzung kurzer Texte in XML-Dokumenten. Es liest die Texte von der Standardeingabe im Format, wie es die Transformation `lt_in.xslt` ausgibt, und erstellt ein XML-Dokument mit den übersetzten Texten, s. `loctext.rng`.

## Optionen

Zu allen Befehlszeilenargumenten gibt es Voreinstellungen.

`--help`

Gibt eine kurze Hilfe mit den Voreinstellungen zu allen möglichen Befehlszeilenargumenten aus.

`--version`

Gibt kurze Hinweise zum Programm und die Version aus.

`--verbose`

Erhöht den Umfang der Meldungen nach STDERR.

`--no_verbose`

Unterdrückt die Ausgabe von Meldungen. Die Optionen `--verbose` und `--no_verbose` werden der Reihe nach ausgewertet.

`--out` *`OUT`*

*`OUT`* ist der Pfad der Ausgabedatei.

`--trname` *`TRNAME`*

*`TRNAME`* ist die Kennung des Übersetzungsprogramms. Die automatische Übersetzung beschreibe ich im Zusammenhang mit der Lokalisierung der website.

`--srclang` *`SRCLANG`*

Ein zu übersetzender Text kann in mehreren Sprachen vorliegen. Wenn er in der Sprache mit der Kennung *`SRCLANG`* existiert, dann ist diese Sprache die Ausganssprache für Übersetzungen. Andernfalls wird eine der existierenden Sprachen ausgewählt.

`--tgtlang` *`TGTLANG`*

*`TGTLANG`* ist eine Liste der Kennungen der Zielsprachen. Die Kennungen der Sorachen sind durch Leerzeichen getrennt. Ein Text wird in eine Zielsprache übersetzt, wenn die Eingabe keine Version in der Zielsprache enthält oder wenn die Option `--retrans` genutzt wird und die Zielsprache nicht die Ausgangssprache ist.

`--retrans`

Ein Text, der bereits in einer Zielsprache vorliegt, wird neu übersetzt, falls die Zielsprache nicht die Ausgangssprache ist.

`--no_retrans`

Ein Text, der bereits in einer Zielsprache vorliegt, wird nicht übersetzt.

# Format der Eingabe

Die Eingabe wird zeilenweise verarbeitet. Leerraumzeichen am Anfang und am Ende einer Zeile werden ignoriert. Es gibt die folgenden Arten von Zeilen:

`POS` *`999`*

Ein zu übersetzender Text (mit Versionen in verschiedenen Sprachen) wird durch eine ID oder seine Position im Quelldokument gekenntzeichnet. *`999`* steht für eine Folge von Dezimalziffern. Bis zur nächsten POS-Zeile oder ID-Zeile werden die Textzeilen als verschiedene Sprachversionen des Textes an der Position *`999`* gelesen.

`ID` *`TEXTID`*

*`TEXTID`* ist eine Folge von Großbustaben (`A` bis `Z`), Kleinbuchstaben (`a` bis `z`), Untestrichen (_) und Ziffen (`0` bis `9`).

`LANG` *`SPRACHKENNUNG`*

Die Zeile nach einer *`LANG`*-Zeile ist eine Textzeile mit einem Text in der Sprache *`SPRACHKENNUNG`*.

*`Textzeile`*

Eine Textzeile folgt unmittelbar nach einer *`LANG`*-Zeile. Sie enthält den Text mit der Kennung aus der letzten vorangehenden *`ID`*- oder *`POS`*-Zeile in der Sprache aus der letzten vorangehenden *`LANG`*-Zeile.

`#` *`Kommentarzeile`*

Eine Zeile, die keine Textzeile ist und mit dem Zeichen # anfängt, ist ein Kommentar. Er hat keine Bedeutung für die Übersetzung.

Leerzeile

Eine Zeile, die keine Textzeile ist, kann leer sein. Sie wird ignoriert.

# Benutzte Module

Das Programm benutzt die folgenden Module:

`Herbaer::Readargs` (`Readargs.pm`)

Die Funktion `Herbaer::Readargs::read_args` liest die Befehlszeilen-Argumente.

`Herbaer::Translate`

Im Zusammenhang mit der Lokalisierung der Website beschrieben.

`Herbaer::XMLDataWriter` (Datei `XMLDataWriter.pm`)

Ausgabe von XML-Daten.

# Quelltext

[Beschreibung]

```
#!/usr/bin/perl -w
# maschinelle Übersetzung von Textvarianten
# 2015-12-16 Herbert Schiemann <h.schieman@herbaer.de>
# GPL Version 2 oder neuer


package main;


use utf8 ;
use Herbaer::Readargs ;
use Herbaer::Translate ;
use Herbaer::XMLDataWriter ;


binmode (STDIN,  ":utf8") ;
binmode (STDOUT, ":utf8") ;
binmode (STDERR, ":utf8") ;


# Hash der Kommandozeilen-Argumente
my $args = {
   "[cnt]verbose" => 1,
   "out"          => "-",       # Ausgabedatei oder - für STDOUT
   "trname"       => "google",  # Übersetzer
   "srclang"      => "de",      # Default-Quellsprache
   "tgtlang"      => "zh",      # Liste der Zielsprachen
   "[cnt]retrans" => undef,     # Fremdsprachige Texte neu übersetzen
};


# gibt die Version nach STDOUT aus
sub version {
   print <<'VERSION' ;
KLEIDER/web/src/kalender/trans.pl
Maschinelle Übersetzung von Textvarianten
2015 Herbert Schiemann <h.schiemann@herbaer.de>
VERSION
}
$args -> {"[sr]version"} = sub { version (); exit 0; };


$args -> {"[sr]help"} = sub {
   version ();
   print_message_with_values (<<'HELP', $args);
trans.pl [Optionen]
--[no_]verbose    Umfang der Meldungen ${[cnt]verbose}
--out OUT         Pfad der Ausgabedatei oder "-" für STDOUT
                  ${out}
--trname TRNAME   Übersetzer ${trname}
--srclang         Default-Quellsprache ${srclang}
--tgtlang         Liste der Zielsprachen, durch Leerzeichen getrennt ${tgtlang}
--[no_]retrans    Fremdsprachige Texte neu übersetzen ${[cnt]retrans}
HELP
   exit 0;
};


read_args ($args);


my $data = {};


read_data ($args, $data);
translate ($args, $data);
write_data ($args, $data);
exit 0;
```

```
sub read_data {
   my ($args, $data) = @_ ;
   my $verb = $args -> {"[cnt]verbose"};
   my $line;                       # Zeile der Eingabe
   my $istext = 0;                 # ist die nächste Zeile eine Textzeile?
   my $tv = {};                    # Hash der Quell-Sprachvarianten zu einer ID
   my $lang = $args -> {"srclang"}; # Quellsprache
   my $lineno = 0;                 # Zeilennummer
   my $src = {};
   $data -> {"src"} = $src;
   while (defined ($line = <STDIN>)) {
      ++$lineno;
      $line =~ s/^\s+// ;
      $line =~ s/\s+$// ;
      if ($istext) {
         $tv -> {$lang} = $line;
         $istext = 0;
      }
      elsif (
         $line =~ /^POS\s+(\d+)\s*$/ or
         $line =~ /^ID\s+([a-zA-Z_0-9])+/
         ) {
         $tv = {};
         $src -> {"$1"} = $tv;
      }
      elsif ( $line =~ /^LANG\s+(\S+)\s*$/ ) {
         $lang = $1;
         $istext = 1;
      }
      elsif ( $line =~ /^#/ or !$line ) {
      }
      else {
         print STDERR "Fehler in Eingabezeile $lineno\n$line\n" if $verb;
      }
   }
} # read_data


sub translate {
   my ($args, $data) = @_ ;

   my $trans = new Herbaer::Translate ($args -> {"trname"}) ;
   if (! $trans) {
      print STDERR "Kann Übersetzer " . $args -> {"trname"} . " nicht laden\n";
      exit 1;
   }

   my $verb    = $args -> {"[cnt]verbose"};
   my $retrans = $args -> {"[cnt]retrans"};
   my $srclang = $args -> {"srclang"};
   my $ll   = [split ("\\s+", $args -> {"tgtlang"})];
   my $trdata  = {};
   my $srcdata = $data -> {"src"};
   $data -> {"trans"} = $trdata;
   my $id;    # Text-ID
   my $tgtl;  # Zielsprache
   my $srcl;  # Quellsprache
   my $hsrc;  # Hash der Quelltexte
   my $htgt;  # Hash der Übersetzungen
   my $tt;    # übersetzter Text
   my $trn;   # Übersetzername
   for $tgtl (@$ll) {
      next unless $tgtl;
      while ( ($id, $hsrc) = each (%$srcdata) ) {
         next if $hsrc -> {$tgtl} && !$retrans;
         $srcl = $srclang;
         if (! $hsrc -> {$srcl}) {
            ( $srcl ) = keys %$hsrc;
         }
         next if $srcl eq $tgtl;
         $htgt = $trdata -> {$id} //= {};
         $tt  = $trans -> translate ($hsrc -> {$srcl}, $srcl, $tgtl);
         $trn = $trans -> translator_name ();
         $htgt -> {$tgtl} = {
            "text"   => $tt,
            "trname" => $trn,
         };
      }
   }
} # translate


sub write_data {
   my ($args, $data) = @_ ;
   my $verb = $args -> {"[cnt]verbose"};
   my $out  = $args -> {"out"};
   my $opt = {
      "%g" => ["g", "v", '@id', "WRITE_EMPTY"],
      "%v" => ["v", "d", '@l'],
   };
   my $writer = Herbaer::XMLDataWriter -> new ($opt);
   $writer -> open ($out, "utf-8", "http://herbaer.de/xmlns/20151212/loctext/", "");
   $writer -> write ("g", {}, $data -> {"trans"});
   $writer -> close ();
} # write_data
```

# lt_merge.xslt

[Quelltext]

# Allgemeines

Übersetzte Texte einfügen

Diese Transformation fügt weitere Sprachversionen in `lt:v`-Elemente ein.

# Namensräume

Die Namensraum-Präfixe, die aus dem erzeugten Dokument ausgeschlossen sind, sind durch einen Stern (*) in der ersten Spalte gekennzeichnet.

| | Präfix | Namensraum |
|---|---|---|
| | xml | http://www.w3.org/XML/1998/namespace |
| | (default) | http://herbaer.de/xmlns/20151212/loctext/ |
| * | lt | http://herbaer.de/xmlns/20151212/loctext/ |
| * | d | http://herbaer.de/xmlns/20051201/doc |
| | xsl | http://www.w3.org/1999/XSL/Transform |

# Eingebundene Stylesheets

## /pool/txt.xslt - Hilfsvorlagen zur Ausgabe und Verarbeitung von Text

Zeilenende: $txt.break

# Parameter

## Parameter p_fnnew

Dateipfad des Dokuments mit den neuen Sprachversionen

Select: ''

Der Parameter wird in den folgenden Toplevel-Elementen benutzt:

Variable g_newlang

## Parameter p_replace

Werden existierende Sprachvarianten ersetzt? Mögliche Werte: `replace` oder `keep`

Select: 'replace'

Der Parameter wird in den folgenden Toplevel-Elementen benutzt:

Muster-Vorlage lt:v

# Globale Variable

## Variable g_newlang

Das Wurzelelement der neuen Sprachversionen

Select: document($p_fnnew)/lt:g

Verwendete globale Parameter oder Variable:

Parameter p_fnnew

Die Variable wird in den folgenden Toplevel-Elementen benutzt:

Muster-Vorlage lt:v

# Muster-Vorlagen (matching templates)

## Muster-Vorlage /

Die Wurzel

## Muster-Vorlage *

Elemente werden kopiert

## Muster-Vorlage comment() | processing-instruction() | text()

Kommentare, Verarbeitungsanweisungen und Text werden kopiert

## Muster-Vorlage @*

Attribute werden kopiert

## Muster-Vorlage lt:v

Verschiedene Sprachversionen eines Textes

Verwendete globale Parameter oder Variable:

Parameter p_replace
Variable g_newlang

## Muster-Vorlage lt:t

### Parameter

nt

Existierende Sprachvarianten werden kopiert

## Muster-Vorlage lt:d

### Parameter

t

Neue Sprachvarianten werden als `lt:t`-Element eingefügt

# Quelltext

[Beschreibung]

```
<?xml version="1.0" encoding="utf-8"?>
<?xml-stylesheet href="/pool/xslt_ht.xslt" type="application/xml"?>
<!--
  Übersetzte Texte einfügen
  2015 Herbert Schiemann <h.schiemann@herbaer.de>
  Borkener Str. 167, 46284 Dorsten, Germany
  GPL Version 2 oder neuer
  Jede Gewährleistung ist ausgeschlossen
-->
<xsl:stylesheet
  xmlns:xsl = "http://www.w3.org/1999/XSL/Transform"
  xmlns:d   = "http://herbaer.de/xmlns/20051201/doc"
  xmlns:lt  = "http://herbaer.de/xmlns/20151212/loctext/"
  xmlns     = "http://herbaer.de/xmlns/20151212/loctext/"
  exclude-result-prefixes = "d lt"
  version   = "1.0"
>
<xsl:param name = "p_fnnew" select  = "''"/>


<xsl:param name = "p_replace" select = "'replace'"/>


<xsl:variable name = "g_newlang" select = "document($p_fnnew)/lt:g"/>


<xsl:include href = "/pool/txt.xslt"/>


<xsl:template match = "/">
  <xsl:apply-templates select = "comment() | processing-instruction() | *"/>
</xsl:template>


<xsl:template match = "*">
  <xsl:copy>
    <xsl:copy-of select = "@*"/>
    <xsl:apply-templates select = "comment() | processing-instruction() | text() | *"/>
  </xsl:copy>
</xsl:template>


<xsl:template match = "comment() | processing-instruction() | text()">
  <xsl:copy-of select = "."/>
</xsl:template>


<xsl:template match = "@*">
  <xsl:copy-of select = "."/>
</xsl:template>


<xsl:template match = "lt:v">
  <xsl:variable name = "id">
    <xsl:choose>
      <xsl:when test = "@id">
        <xsl:value-of select = "@id"/>
      </xsl:when>
      <xsl:otherwise>
        <xsl:value-of select = "count (preceding::lt:v) + 1"/>
      </xsl:otherwise>
    </xsl:choose>
  </xsl:variable>
  <xsl:variable name = "nt" select = "$g_newlang/lt:v[@id = $id]"/>
  <v>
    <xsl:apply-templates select = "@*"/>
    <xsl:choose>
      <xsl:when test = "$p_replace = 'keep'">
        <xsl:apply-templates select = "lt:t"/>
        <xsl:apply-templates select = "$nt/lt:d">
          <xsl:with-param name = "t" select = "."/>
        </xsl:apply-templates>
      </xsl:when>
      <xsl:otherwise>
        <xsl:apply-templates select = "lt:t">
          <xsl:with-param name = "nt" select = "$nt"/>
        </xsl:apply-templates>
        <xsl:apply-templates select = "$nt/lt:d"/>
      </xsl:otherwise>
    </xsl:choose>
  </v>
</xsl:template>
```

```
<xsl:template match = "lt:t">
  <xsl:param name = "nt"/>
  <xsl:variable name = "l" select = "@l"/>
  <xsl:choose>
    <xsl:when test = "$nt and $nt/lt:d[@l = $l]"/>
    <xsl:otherwise>
      <!--
      <xsl:copy-of select = "."/>
      -->
      <t>
        <xsl:apply-templates select = "@*"/>
        <xsl:apply-templates select = "text()"/>
      </t>
      <xsl:value-of select = "$txt.break"/>
    </xsl:otherwise>
  </xsl:choose>
</xsl:template>


<xsl:template match = "lt:d">
  <xsl:param name = "t"/>
  <xsl:variable name = "l" select = "@l"/>
  <xsl:choose>
    <xsl:when test = "$t and $t/lt:t[@l = $l]"/>
    <xsl:otherwise>
      <t l="{@l}" tr="{lt:trname}"><xsl:value-of select = "lt:text"/></t>
      <xsl:value-of select = "$txt.break"/>
    </xsl:otherwise>
  </xsl:choose>
</xsl:template>


</xsl:stylesheet>
```

# lt_chktrname.xslt

[Quelltext]

# Allgemeines

Attribut lt:t/@tr entfernen / ersetzen

Diese Transformation ersetzt, entfernt oder ergänzt `lt:t/@tr`-Attribute.

# Namensräume

| Präfix | Namensraum |
|---|---|
| xml | http://www.w3.org/XML/1998/namespace |
| lt | http://herbaer.de/xmlns/20151212/loctext/ |
| d | http://herbaer.de/xmlns/20051201/doc |
| xsl | http://www.w3.org/1999/XSL/Transform |

# Eingebundene Stylesheets

## /pool/list.xslt - Vorlagen zur Arbeit mit Listen

Listenverarbeitung: $list.map

Diese Datei enthält benannte Hilfsvorlagen, die die Einträge einer Liste akkumulieren, zu einer neuen Liste bearbeiten, auf zwei Listen aufteilen oder sortieren. Sie nutzt Vorlage aus `txt.xslt`. Wenn ein Stylesheet diese Datei (`list.xslt`) einbindet, sollte es daher auch `txt.xslt` einbinden.

Eine Liste ist eine Zeichenkette, die durch eine feste Zeichenfolge (die Trennzeichenfolge) in einzelne Listeneinträge zerlegt wird. Einige Vorlagen behandeln die leere Zeichenkette als Listeneintrag so, als gäbe es diesen Eintrag nicht. Ein Listeneintrag sollte also, so wie Listen hier behandelt werden, nicht leer sein.

Die Trennzeichenfolge kann jeder Vorlage für jede Liste als Parameter übergeben werden. Wenn eine Vorlage mit mehreren Listen arbeitet, können die Listen verschiedene Trennzeichenfolgen verwenden. Voreingestellt ist ein einzelnes Leerzeichen.

Die Listeneinträge werden von 1 an gezählt (Position eines Listeneintrags).

Konkrete Funktionen für einzelne Listeneinträge werden ausgeführt, indem die Vorlagen im Modus `list.apply` auf einen Parameter ("Funktional") angewandt werden. Typischerweise ist der Parameter eine Vorlage, die auf sich selbst im Modus `list.apply` passt. Zu jedem "Funktional"-Parameter gibt es einen weiteren Parameter (meist mit dem Namen "*param*", der an die Vorlagen im Modus `list.apply` übergeben wird.

## /pool/txt.xslt - Hilfsvorlagen zur Ausgabe und Verarbeitung von Text

# Parameter

## Parameter p_replace

Die Liste der Ersetzungen im Format *ENTHALTEN*:*NEUER_ATTWERT*(#*ENTHALTEN*:*NEUER_ATTWERT*)*. Wenn der Wert des Attributs lt:t/@tr in der Eingabe die Zeichenfolge *ENTHALTEN* enthält, wird *NEUER_ATTWERT*

der neue Wert des Attributs in der Ausgabe. Wenn keine der Zeichenfolgen *ENTHALTEN* im Attributwert in der Eingabe enthalten ist oder die Zeichenfolge *NEUER_ATTWERT* leer ist, wird das Attribut entfernt. Auch die Zeichenfolge *ENTHALTEN* kann leer sein.

Select: 'google:Google Translate#:Google Translate'

Der Parameter wird in den folgenden Toplevel-Elementen benutzt:

Muster-Vorlage lt:t/@tr

### Parameter p_default

Der Default-Wert des Attributs lt:t/@tr. Wenn das Attribut in der Eingabe fehlt und dieser Wert nicht die leere Zeichenkette ist, wird ein Attribut mit diesem Wert eingefügt.

Select: ''

Der Parameter wird in den folgenden Toplevel-Elementen benutzt:

Muster-Vorlage lt:t

## Globale Variable

### Variable g_map_trvalue

Vorlage zur Auswahl des passenden Ersetzungswertes

Select: document('')/xsl:stylesheet/xsl:template[@name = 'list.map.trvalue']

Die Variable wird in den folgenden Toplevel-Elementen benutzt:

Muster-Vorlage lt:t/@tr

## Muster-Vorlagen (matching templates)

### Muster-Vorlage /

Die Wurzel

### Muster-Vorlage *

Elemente werden kopiert

### Muster-Vorlage comment() | processing-instruction() | text()

Kommentare, Verarbeitungsanweisungen und Text werden kopiert

### Muster-Vorlage @*

Attribute werden kopiert

### Muster-Vorlage lt:t

Eine Sprachvariante

Verwendete globale Parameter oder Variable:

Parameter p_default

## Muster-Vorlage lt:t/@tr

Kennung des Übersetzers

Aufgerufene benannte Vorlagen:

list.map

Verwendete globale Parameter oder Variable:

Parameter p_replace
Variable g_map_trvalue

## Muster-Vorlage xsl:template [@name = 'list.map.trvalue' and @mode = 'list.apply' and @match], list.apply

Name: list.map.trvalue

### Parameter

item

ein Ersetzungs-Paar ENTHALTEN:NEUER_ATTWERT

param

das Attribut @tr

Passende neue Attributwerte herausfiltern

# Modus

## Modus list.apply

Die folgenden Vorlagen implementieren den Modus list.apply:

Muster-Vorlage xsl:template [@name = 'list.map.trvalue' and @mode = 'list.apply' and @match], list.apply

# Quelltext

[Beschreibung]

```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet href="/pool/xslt_ht.xslt" type="application/xml"?>
<!--
  Attribut lt:t/@tr entfernen / ersetzen
  2017 Herbert Schiemann <h.schiemann@herbaer.de>
  Borkener Str. 167, 46284 Dorsten, Germany
  GPL Version 2 oder neuer
  Jede Gewährleistung ist ausgeschlossen
-->
<xsl:stylesheet
  xmlns:xsl = "http://www.w3.org/1999/XSL/Transform"
  xmlns:d   = "http://herbaer.de/xmlns/20051201/doc"
  xmlns:lt = "http://herbaer.de/xmlns/20151212/loctext/"
  version   = "1.0"
>
<xsl:param name = "p_replace" select = "'google:Google Translate#:Google Translate'"/>


<xsl:param name = "p_default" select = "''"/>


<xsl:include href = "/pool/list.xslt"/>
<xsl:include href = "/pool/txt.xslt"/>


<xsl:variable
    name   = "g_map_trvalue"
    select = "document('')/xsl:stylesheet/xsl:template[@name = 'list.map.trvalue']"
/>


<xsl:template match = "/">
  <xsl:apply-templates select = "comment() | processing-instruction() | *"/>
</xsl:template>


<xsl:template match = "*">
  <xsl:copy>
    <xsl:copy-of select = "@*"/>
    <xsl:apply-templates select = "comment() | processing-instruction() | text() | *"/>
  </xsl:copy>
</xsl:template>


<xsl:template match = "comment() | processing-instruction() | text()">
  <xsl:copy-of select = "."/>
</xsl:template>


<xsl:template match = "@*">
  <xsl:copy-of select = "."/>
</xsl:template>


<xsl:template match = "lt:t">
  <xsl:copy>
    <xsl:apply-templates select = "@*"/>
    <xsl:if test = "not (@tr) and string-length ($p_default) &gt; 0">
      <xsl:attribute name = "tr">
        <xsl:value-of select = "$p_default"/>
      </xsl:attribute>
    </xsl:if>
    <xsl:apply-templates select = "text()"/>
  </xsl:copy>
</xsl:template>


<xsl:template match = "lt:t/@tr">
  <xsl:variable name = "list">
    <xsl:call-template name = "list.map">
      <xsl:with-param name = "list"   select = "$p_replace"/>
      <xsl:with-param name = "mapper" select = "$g_map_trvalue"/>
      <xsl:with-param name = "param"  select = "."/>
      <xsl:with-param name = "sep"    select = "'#'"/>
    </xsl:call-template>
  </xsl:variable>
  <xsl:choose>
    <xsl:when test = "string-length ($list) &gt; 0">
      <xsl:variable name = "nv" select = "substring-before ($list, '#')"/>
      <xsl:if test = "string-length ($nv) &gt; 0">
        <xsl:attribute name = "tr">
          <xsl:value-of select = "$nv"/>
        </xsl:attribute>
      </xsl:if>
    </xsl:when>
    <xsl:otherwise>
      <xsl:copy-of select = "."/>
    </xsl:otherwise>
  </xsl:choose>
</xsl:template>
```

```
<xsl:template
  name  = "list.map.trvalue"
  mode  = "list.apply"
  match =
    "xsl:template [@name = 'list.map.trvalue' and @mode = 'list.apply' and @match]"
>
  <xsl:param name = "item"/>
  <xsl:param name = "param"/>
  <xsl:variable name = "ent" select = "substring-before ($item, ':')"/>
  <xsl:variable name = "nv"  select = "substring-after ($item, ':')"/>
  <xsl:if test = "contains ($param, $ent)">
    <xsl:value-of select = "concat ($nv, '#')"/>
  </xsl:if>
</xsl:template>


</xsl:stylesheet>
```

# kal_addkeywords.xslt

[Quelltext]

# Allgemeines

Schlüsselwörter zu Kalenderbildern hinzufügen

Diese Transformation fügt „`Kalender`" als Stichwort für Suchmaschinen in den angebotenen Sprachen hizu.

# Namensräume

Die Namensraum-Präfixe, die aus dem erzeugten Dokument ausgeschlossen sind, sind durch einen Stern (*) in der ersten Spalte gekennzeichnet.

| | **Präfix** | **Namensraum** |
|---|---|---|
| | xml | http://www.w3.org/XML/1998/namespace |
| | (default) | http://herbaer.de/xmlns/20151211/kalenderbilder/ |
| * | lt | http://herbaer.de/xmlns/20151212/loctext/ |
| * | l | http://herbaer.de/xmlns/20141210/localization |
| * | kb | http://herbaer.de/xmlns/20151211/kalenderbilder/ |
| * | h | http://herbaer.de/xmlns/20121015/hash |
| | ht | http://www.w3.org/1999/xhtml |
| * | d | http://herbaer.de/xmlns/20051201/doc |
| | xsl | http://www.w3.org/1999/XSL/Transform |

# Parameter

## Parameter p_shortids

Pfad des XML-Dokuments mit den kurzen IDs der Platzhalter

Select: 'shortids.xml'

Der Parameter wird in den folgenden Toplevel-Elementen benutzt:

Parameter g_kalid

## Parameter p_docroot

Document Root - Verzeichnis des Webservers

Select: 'docroot/'

Der Parameter wird in den folgenden Toplevel-Elementen benutzt:

Muster-Vorlage lt:t, meta

## Parameter g_kalid

Textkennung für `kalender`

Select: document($p_shortids)/h:hash/h:value [@key = 'kalender']

Verwendete globale Parameter oder Variable:

Parameter p_shortids

Der Parameter wird in den folgenden Toplevel-Elementen benutzt:

Muster-Vorlage lt:t, meta

# Muster-Vorlagen (matching templates)

## Muster-Vorlage /

## Muster-Vorlage kb:kalenderbilder

Zum Wurzelelement wird ein HTML-Meta-Element mit Schlüsselwörtern hinzugefügt.

Verwendete Modus:

meta

## Muster-Vorlage lt:t, meta

Stichwort in einer angebotenen Sprache

Verwendete globale Parameter oder Variable:

Parameter p_docroot
Parameter g_kalid

## Muster-Vorlage *

Elemente werden absteigend kopiert.

## Muster-Vorlage @*

Attribute werden kopiert.

## Muster-Vorlage processing-instruction()

Verarbeitungsanweisungen werden kopiert.

# Modus

## Modus meta

Die folgenden Vorlagen implementieren den Modus meta:

Muster-Vorlage lt:t, meta

Der Modus meta wird in den folgenden Stylesheet-Elementen benutzt:

Muster-Vorlage kb:kalenderbilder

# Quelltext

[Beschreibung]

```
<?xml version="1.0" encoding ="utf-8"?>
<?xml-stylesheet href="/pool/xslt_ht.xslt" type="application/xml"?>
<!--
  Schlüsselwörter zu Kalenderbildern hinzufügen
  2015 Herbert Schiemann <h.schiemann@herbaer.de>
  Borkener Str. 167, 46284 Dorsten, Germany
  GPL Version 2 oder neuer
  Jede Gewährleistung ist ausgeschlossen
-->
<xsl:stylesheet
  xmlns:xsl = "http://www.w3.org/1999/XSL/Transform"
  xmlns:d   = "http://herbaer.de/xmlns/20051201/doc"
  xmlns:ht  = "http://www.w3.org/1999/xhtml"
  xmlns:h   = "http://herbaer.de/xmlns/20121015/hash"
  xmlns:kb  = "http://herbaer.de/xmlns/20151211/kalenderbilder/"
  xmlns:l   = "http://herbaer.de/xmlns/20141210/localization"
  xmlns:lt  = "http://herbaer.de/xmlns/20151212/loctext/"
  xmlns     = "http://herbaer.de/xmlns/20151211/kalenderbilder/"
  exclude-result-prefixes = "d lt h kb l"
  version   = "1.0"
  xml:lang  = "de"
>
<xsl:param name = "p_shortids" select = "'shortids.xml'"/>


<xsl:param name = "p_docroot" select = "'docroot/'"/>


<xsl:param name = "g_kalid"
  select = "document($p_shortids)/h:hash/h:value [@key = 'kalender']"
/>


<xsl:template match = "/">
  <xsl:apply-templates select = "* | processing-instruction()"/>
</xsl:template>


<xsl:template match = "kb:kalenderbilder">
  <kalenderbilder>
    <xsl:apply-templates select = "@*"/>
    <xsl:if test = "not (ht:meta [@name = 'keywords'])">
      <xsl:choose>
        <xsl:when test = "kb:t/lt:v/lt:t">
          <xsl:apply-templates select = "kb:t/lt:v/lt:t" mode = "meta"/>
        </xsl:when>
        <xsl:when test = "kb:y">
          <ht:meta name="keywords" content="Kalender {kb:y}" lang="de"/>
        </xsl:when>
        <xsl:otherwise>
          <ht:meta name="keywords" content="Kalender" lang="de"/>
        </xsl:otherwise>
      </xsl:choose>
    </xsl:if>
    <xsl:apply-templates select = "*"/>
  </kalenderbilder>
</xsl:template>


<xsl:template match = "lt:t" mode = "meta">
  <xsl:variable name = "lang" select = "@l"/>
  <xsl:variable name = "cont">
    <xsl:value-of select = "
      document(concat($p_docroot, '/local/local.xml.', $lang, '.'))
      /l:localization/l:t[@id=$g_kalid]
    "/>
    <xsl:if test = "/kb:kalenderbilder/kb:y">
      <xsl:text> </xsl:text>
      <xsl:value-of select = "/kb:kalenderbilder/kb:y"/>
    </xsl:if>
  </xsl:variable>
  <ht:meta name="keywords" content="{$cont}" lang="{$lang}"/>
</xsl:template>


<xsl:template match = "*">
  <xsl:copy>
    <xsl:apply-templates select = "@* | text() | *"/>
  </xsl:copy>
</xsl:template>


<xsl:template match = "@*">
  <xsl:copy-of select = "."/>
</xsl:template>


<xsl:template match = "processing-instruction()">
  <xsl:copy-of select = "."/>
</xsl:template>


</xsl:stylesheet>
```

# kalender

[Quelltext]

## Übersicht

`kalender --help|--version`

`kalender [ --verbose | --no_verbose ] [ --overwrite | --no_overwrite ] [ --translate | --no_translate ] [ --trname` *`TRNAME`* `] [ --trreplace` *`TRREPLACE`* `] [ --trlang` *`TRLANG`* `] [ --basedir` *`BASEDIR`* `][ --docroot` *`DOCROOT`* `][ --kaldir` *`KALDIR`* `][ --datadir` *`DATADIR`* `][ --srcbase` *`SRCBASE`* `][ --srcdir` *`SRCDIR`* `][ --xsltpool` *`XSLTPOOL`* `][ --tempdir` *`TEMPDIR`* `][ --keeptmp | --no_keeptmp ][ [ --tree | --kalender | --xslt | --mkhelp | --base | --langbase | --index | --htaccess ]…`
`| [ --no_tree | --no_kalender | --no_xslt | --no_mkhelp | --no_base | --no_index | --no_htaccess ]… ]`

## Optionen

`--help`

Gibt eine kurze Hilfe aus und zeigt die aktuellen Einstellungen an.

`--version`

Gibt kurze Hinweise zum Programm und die Version aus.

`--verbose`

Meldungen über den Programmablauf werden nach STDOUT ausgegeben.

`--no_verbose`

Diese Option hebt die Wirkung der Option `--verbose` auf.

`--overwrite`

Existierende Dateien werden überschrieben.

`--no_overwrite`

Diese Option hebt die Wirkung der Option `--overwrite` auf: existierende Dateien bleiben erhalten.

`--translate`

Texte in den Bildauswahl-Dateien (die Titel) werden in die fehlenden Sprachen übersetzt. Die Übersetzung ist Teil der Aktion `--kalender`.

Die Übersetzung ist nur möglich, wenn die Dateien `trans.pl`, `lt_in.xslt`, `lt_merge.xslt` und `lt_chktrname.xslt` im Verzeichnis *`SRCDIR`* existieren. Wenn die Kennung des Übersetzungsmoduls (*`TRNAME`*) die Zeichenkette `pipe` enthält, wird ferner vorausgesetzt, dass die Skripte *`SRCBASE`*`/localization/pipe_srv.pl` und *`SRCBASE`*`/localization/pipe_srv_stop.pl` ausgeführt werden können.

`--no_translate`

Diese Option hebt die Wirkung der Option `--translate` auf. Die Titel der Bildauswahlen werden nicht in weitere Sprachen übersetzt.

`--trname` *TRNAME*

*TRNAME* ist die Kennung des Übersetzers. Sie wird im Zusammenhang mit der Lokalisierung der Website beschrieben.

`--trreplace` *TRREPLACE*

Die Transformation `lt_merge.xslt` fügt die übersetzten Titel in eine Kalenderbilddatei ein, zusammen mit den Kennungen der Übersetzer. In den Kalenderbilddateien, die der Webserver liefert, sind die Kennungen der Übersetzer durch andere Namen ersetzt. Die Zeichenkette *TRREPLACE* beschreibt diese Ersetzung.

Die Zeichenkette *TRREPLACE* wird als Parameter *p_replace* an die Transformation `lt_chktrname.xslt` übergeben. Dort ist der Aufbau des Parameterwertes beschrieben.

`--trlang` *TRLANG*

*TRLANG* ist eine Liste der Zielsprachen, in die die Titel der Bildauswahlen übersetzt werden. Die Kennungen der Zielsprachen werden durch ein Leerzeichen getrennt. In der Voreinstellung sind die Zielsprachen alle Sprachen, in denen die Lokalisierungsdatei *DOCROOT*`/local/local.xml` verfügbar ist.

`--basedir` *BASEDIR*

Der Pfad des Basisverzeichnisses zur Website. Er beeinflusst die voreingestellten Pfade der anderen Verzeichnisse.

`--docroot` *DOCROOT*

Das „Document root”-Verzeichnis des Webservers.

`--kaldir` *KALDIR*

*KALDIR* ist das Unterverzeichnis von *DOCROOT* mit den Daten zu den Kalendern (oder ein „Spiegelverzeichnis”).

`--datadir` *DATADIR*

*DATADIR* ist das Verzeichnis der Quelldaten zu den Kalendern Es enthält die Bildauswahldateien *DATADIR*`/**/`*BILDAUSWAHL*`.xml` und kann Index-Dateien *DATADIR*`/**/index.xhtml.de` und *DATADIR*`/index.xhtml.de` enthalten.

Die Verzeichnisstruktur spielt hier keine Rolle. Die Bildauswahl-Dateien für Kalender des Jahres *JAHR* liegen im Unterverzeichnis *DATADIR*`/`*JAHR*.

Die Kalender-Grunddaten liegen unter den Dateipfaden *DATADIR*`/base/`*JAHR*`/a.xml.`*LANG*. *LANG* steht für die Kennung der Sprache und des Landes in Kleinbuchstaben, z.B. `fr-ca` für Französisch in Kanada. Die Grunddaten enthalten in diesem Beispiel die gesetzlichen Feiertage in der französischsprachigen kanadischen Provinz Quebec.

Die Datei der Baumstruktur der Kalender ist *DATADIR*`/tree.xml`.

`--srcbase` *SRCBASE*

*SRCBASE* ist das Basisverzeichnis des Skripte zur Website. Hier werden Dateien in den Unterverzeichnissen *SRCBASE*`/localization` (Lokalisierung) *SRCBASE*`/style` (Stil allgemein) benötigt.

Das Verzeichnis *SRCDIR* ist in der Voreinstellung das Unterverzeichnis *SRCBASE*`/kalender`.

`--srcdir` *SRCDIR*

Das Verzeichnis *SRCDIR* enthält die Skripte zu den Kalendern.

`--xsltpool` *XSLTPOOL*

Das Verzeichnis *XSLTPOOL* enthält allgemein benutzte XSLT-Dateien.

`--tempdir` *`TEMPDIR`*

Das Verzeichnis *`TEMPDIR`* enthält Zwischendateien (verdichtete CSS- und Javascript-Dateien, Datei zur Ersetzung von Text-Kennungen), s. `--keeptmp`.

`--keeptmp`

Zwischendateien (verdichtete Versionen von CSS- und Javascript-Dateien, Datei zu Ersetzung von Text-IDs) bleiben erhalten.

`--no_keeptmp`

Diese Option hebt die Wirkung der Option `--keeptmp` auf. Zwischendateien (verdichtete Versionen von CSS- und Javascript-Dateien, Datei zu Ersetzung von Text-IDs) werden gelöscht.

`--tree`

Erzeugt die Datei *`DATADIR`*`/tree.xml` mit der Baumstruktur der Kalender. Diese Datei ist die Grundlage zur Erzeugung der Indexdateien (`--index`) und der Bildauswahldateien (`--kalender`) im Verzeichnis *`DOCROOT`*.

Das Programm `tree.pl` erstellt die Baumstruktur (s. `tree.rng`). Dazu benötigt es weitere Dateien.

Die Transformation `tree_sorttlv.xslt` sortiert anschließend die Knoten der zweiten Ebene unterhalb der obersten Knoten in umgekehrter alphabetischer Reihenfolge ihrer Kennung. (Das neuere Jahr steht über dem älteren Jahr.) Hier hat die Baumstruktur genau einen obersten Knoten "Kalender".

`--kalender`

Erstellt Bildauswahl-Dateien im „Document Root„-Verzeichnis (*`DOCROOT`*). Die Liste der relativen URI der Bildauswahldateien wird aus der Baumstruktur *`DATADIR`*`/tree.xml` bestimmt.

Die folgenden Dateien sind beteiligt:

`lt_in.xslt`

Gibt den Titel einer Bildauswahldatei in verschiedenen Sprachversionen zur Weiterverarbeitung durch `trans.pl` aus.

`trans.pl`

Erzeugt die XML-Datei *`TEMPDIR/TIMESTAMP/PFAD_BILDAUSWAHL`*`.xml.trans` mit den fehlenden Übersetzungen des Titels der Bildauswahldatei *`DATADIR/PFAD_BILDAUSWAHL`*`.xml`.

`lt_merge.xslt`

Fügt die in der Datei *`DATADIR/PFAD_BILDAUSWAHL`*`.xml` fehlenden Sprachversionen des Titels aus der Datei *`TEMPDIR/TIMESTAMP/PFAD_BILDAUSWAHL`*`.xml.trans` ein. Die Ausgabe durchläuft die folgende Transformation `lt_chktrname.xslt`.

`lt_chktrname.xslt`

Ersetzt in der Ausgabe der Transformation `lt_merge.xslt` die Kennungen der Übersetzer durch Namen. Das Ergebnis ist *`TEMPDIR/TIMESTAMP/PFAD_BILDAUSWAHL`*`.xml`.

*`SRCBASE`*`/localization/pipe_srv.pl`

Startet den Dienst zur Übersetzung, falls die Kennung des Übersetzungsmoduls *`TRNAME`* die Zeichenkette `pipe` enthält.

*`SRCBASE`*`/localization/pipe_srv_stop.pl`

Bedeendet den durch *`SRCBASE`*`/localization/pipe_srv.pl` gestarteten Dienst.

*`TEMPDIR/TIMESTAMP`*`/PFAD_BILDAUSWAHL.xml.trans`

Die neuen Übersetzungen eines Bildauswahl-Titels.

*TEMPDIR*/*TIMESTAMP*/PFAD_BILDAUSWAHL.xml

Bildauswahldatei mit neuen Übersetzungen des Titels.

kal_addkeywords.xslt

Fügt das Stichwort „Kalender” für Suchmaschinen in den angebotenen Sprachen zu einer Bildauswahldatei hinzu.

add_sspi.xslt

Fügt, falls nötig, eine xml-stylsheet-Verarbeitungsanweisung ein.

*TEMPDIR*/TIMESTAMP/shortids.xml

Datei mit den kurzen Textkennungen in der Lokalisierungsdatei. Die Transformation kal_addkeywords.xslt braucht das Wort „Kalender” in verschiedenen Sprachen.

Die Erstellung dieser Datei ist im Zusammenhang mit dem allgemeinen Stil der Website beschrieben. Die folgenden Dateien sind beteiligt: style/shortids.pl, style/localization_idlist.xslt und style/local.xml.de.

*DATADIR*/tree.xml

Diese Datei liefert mittels der Transformation tree_files.xslt die relativen URI der Bildauswahldateien.

tree_files.xslt

Erstellt aus der XML-Baumstruktur der Kalender (*DATADIR*/tree.xml) eine Liste der relativen URL der Bildauswahldateien.

kal_addkeywords.xslt

Fügt den Bildauswahldateien das Stichwort „Kalender” in den angebotenen Sprachen hinzu.

*XSLTPOOL*/xml_minimize.xslt

Entfernt Kommentare und unnötige Leerzeichen aus einer Bildauswahldatei.

--xslt

Erstellt die XSLT-Dateien im Verzeichnis *KALDIR*/s: *KALDIR*/s/kal.xslt. und *KALDIR*/s/treelist.xslt. sowie die gzip-komprimierten Varianten. Der Inhalt von CSS- und Javascript-Dateien wird direkt an Stelle von Verweisen eingefügt. Die langen „sprechenden” Kennungen der kurzen sprachabhängigen Texte werden durch kurze IDs ersetzt.

Die Erstellung der XSLT-Dateien ist im Zusammenhang mit dem Stil im allgemeinen beschrieben.

--mkhelp

Erzeugt die Hilfe-Dateien *KALDIR*/s/kal_help.xhtml.*, wie im Zusammenhang mit dem Stil im allgemeinen beschrieben.

--base

Erzeugt die Kalendergrunddaten *KALDIR*/b/*.

Grundlage sind die Dateien *KENNUNG*.txt im Verzeichnis *DATADIR*/feiertage und dessen Unterverzeichnissen. Unterverzeichnisse werden rekursiv verarbeitet. *KENNUNG* steht für einen Ländercode oder einen Sprache-/Land-Code wie de-at.

Unterverzeichnisnamen der Form 20[0-9][0-9] bezeichnen ein Kalenderjahr. Wenn es im Pfad keinen solchen Unterverzeichnisnamen gibt, hängt das Jahr vom aktuellen Kalenderjahr ab. In den Monaten Januar bis Oktober ist das Jahr das aktuelle Kalenderjahr, in den Monaten November und Dezember das Folgejahr.

Das Programm base.pl im Verzeichnis *SRCDIR* erzeugt im entsprechenden Unterverzeichnis von *KALDIR*/b die Datei a.xml mit den Grunddaten des Jahres ohne Feiertage.

Aus einer Datei *DATADIR*/feiertage/**/*KENNUNG*.txt erzeugt base.pl eine Grunddaten-XHML-Datei. Anschließende Transformationen erzeugen daraus die Datei *KALDIR*/b/**/*KENNUNG*.xml. und die gzip-komprimierte Version. Die langen Kennungen der sprachabhängigen Texte (Monatsnamen) in den Quelldateien *DATADIR*/base/* werden durch kurze IDs ersetzt, Kommentare und unnötige Leerzeichen und unnötige Namensraumknoten werden entfernt.

Unter anderem sind die folgenden Dateien beteiligt. Die Einzelheiten sind im Zusammenhang mit dem Stil im allgemeinen erklärt.

SRCDIR/base.pl

Erzeugt aus einer Textdatei mit den Feiertagen eine XHTML-Grunddaten-Datei.

TEMPDIR/TIMESTAMP/shortids.xml

Kurze Kennungen der sprachabhängigen Texte

SRCBASE/style/localization_shortids.xslt

Ersetzt die langen „sprechenden” Kennungen durch die kurzen Kennungen

XSLTPOOL/xml_minimize.xslt

Entfernt Kommentare und unnötige Leerzeichen.

SRCBASE/style/rmxmlns.pl

Entfernt unnötige Definitionen von Namensraum-Präfixen.

--langbase

Jede angebotene Sprachen wird einem Land zugeordnet. *SPRACHE* ist die Kennung einer Sprache, *LAND* die Kennung des zugeordneten Landes. Wenn die Datei *KALDIR*/b/*/*SPRACHE-LAND*.xml. existiert, dann wird diese Datei nach *KALDIR*/b/*/a.*SPRACHE*.xml. kopiert, und die gzip-komprimierte Datei *KALDIR*/b/*/*SPRACHE-LAND*.xml. wird nach *KALDIR*/b/*/a.*SPRACHE*.xml.gz kopiert. Wenn andernfalls die Datei *KALDIR*/b/*/*LAND*.xml. existiert, dann wird diese Datei nach *KALDIR*/b/*/a.*SPRACHE*.xml. kopiert, und die gzip-komprimierte Datei *KALDIR*/b/*/*LAND*.xml.gz wird nach *KALDIR*/b/*/a.*SPRACHE*.xml.gz kopiert.

Die Tansformation langcodes_cmd.xslt liest aus der Datei langcodes.dbd die Paare von *SPRACHE* und *LAND*.

--index

Das Verzeichnis *KALDIR* und dessen Unterverzeichnisse (mit Ausnahme der Unterverzeichnisse s (Stil) und b (Kalendergrunddaten) enthalten Index-Dateien index.xhtml.*LANG*. und index.xhtml.*LANG*.gz in verschiedenen Sprachen.

Die Liste der Unterverzeichnisse wird von der Transformation tree_cmd.xslt aus der Baumstruktur *DATADIR*/tree.xml erzeugt.

Zu einem Unterverzeichnis *KALDIR/SUBDIR* werden zuerst Index-Dateien aus den Dateien *DATADIR/SUBDIR*/index.xhtml.*LANG* erzeugt. Falls dann keine deutschsprachige Indexdatei *KALDIR/SUBDIR*/index.xhtml.de oder *KALDIR/SUBDIR*/index.xhtml.de. existiert, wird die deutschsprachige Index-Datei mit der Transformation tree_ht.xslt aus der Baumstruktur *DATADIR*/tree.xml erzeugt. Die Transformation xhtml_setlinks.xslt fügt Verweise auf das Icon der Website und CSS-Regeln ein. Die Transformation xhtml_settarget.xslt läßt die Ziele der Verweise in einem neuen Browser-Fenster erscheinen.

`--htaccess`

Erzeugt die Datei *KALDIR*`/b/.htaccess`: Das Programm *SRCBASE*`/style/clean_config.pl` entfernt aus der Quelldatei `htaccess` Kommentarzeilen und Leerzeilen.

`--no_*`

Wenn keine der „Aktionsoptionen” `--tree`, `--kalender`, `--xslt`, `--mkhelp`, `--base`, `--langbase`, `--index`, `--htaccess` genutzt wird, können die Optionen mit dem Präfix `no_`" (`--no_tree`, `--no_kalender`, `--no_xslt`, `--no_mkhelp`, `--no_base`, `--no_langbase`, `--no_index` und `--no_htaccess`) genutzt werden. Diese schließen die zugehörige Aktion aus. Alle nicht ausgeschlossenen Aktionen werden ausgeführt.

# Beschreibung

Das Skript `kalender` unterstützt die Pflege der Kalender der Website. Es stützt sich auch auf Dateien in den Verzeichnissen *SRCBASE*`/localization` und *SRCBASE*`/style`, die in anderen Zusammenhängen beschrieben werden.

# Quelltext

[Beschreibung]

```
#!/bin/bash
# -*- coding:utf-8 -*-
# Kalender auf der Website http://kleider.herbaer.de
# 2017-05-31 Herbert Schiemann <h.schiemann@herbaer.de>


# 2020-04-12 proc_langbase Kopien statt SymLinks
# 2020-12-17 Länderkennung in Großbuchstaben, Jahr-Vorgabe.
# 2022-01-29 Verbesserung lang-COUNTR in proc_langbase


# Zähler, Variable, Aktionen
declare_vars ()
{
   # Zähler
   g_counters="  \
      verbose    \
      overwrite  \
      translate  \
      keeptmp    ";

   # Variable
   g_variables=" \
      basedir    \
      docroot    \
      kaldir     \
      datadir    \
      srcbase    \
      srcdir     \
      xsltpool   \
      tempdir    \
      trname     \
      trreplace  \
      trlang     ";

   # Aktionen
   g_actions="   \
      tree       \
      kalender   \
      xslt       \
      mkhelp     \
      base       \
      langbase   \
      index      \
      htaccess   ";
} # declare_vars


# setzt Vorgabe-Werte
set_defaults ()
{
   local b=$(realpath $0);
   b=${b%/src/kalender/kalender};
   [[ -n "$verbose"       ]] || verbose=1 ;
   [[ -n "$overwrite"     ]] || overwrite=0 ;
   [[ -n "$translate"     ]] || translate=1 ;
   [[ -n "$keeptmp"       ]] || keeptmp=0 ;
   [[ -n "$basedir"       ]] || basedir=$b ;
   [[ -n "$docroot"       ]] || docroot=$basedir/docroot ;
   [[ -n "$kaldir"        ]] || kaldir=$docroot/kal ;
   [[ -n "$datadir"       ]] || datadir=$basedir/kalender ;
   [[ -n "$srcbase"       ]] || srcbase=$basedir/src ;
   [[ -n "$srcdir"        ]] || srcdir=$srcbase/kalender ;
   [[ -n "$xsltpool"      ]] || xsltpool=/pool ;
   [[ -n "$tempdir"       ]] || tempdir=$basedir/temp ;
   [[ -n "$trname"        ]] || trname=pipe_nscnr_google ;
   [[ -n "$trreplace"     ]] || trreplace=":Google Translate" ;
   [[ -n "$trlang"        ]] || set_default_trlang ;
   [[ -n "$trlang"        ]] || trlang=" \
     eu ca zh hr cs da nl en eo et fi fr gl el hu is ga it ja ko la lv lt  ms mt \
     no pl pt ro sk sl es sv th vi ";
} # set_defaults


# setzt die Default-Sprachen
set_default_trlang () {
   trlng="" ;
   local f;
   local l;
   for f in $docroot/local/local.xml.*.; do
      l=${f#$docroot/local/local.xml.};
      l=${l%%.*};
      [[ $l == $srclang ]] || trlang="$trlang $l";
   done;
} # set_default_trlang
```

```
# Zeigt eine kurze Hilfe an
show_help ()
{
   local cmd=${0#*/} ;
   set_defaults ;
   cat << .HELP ;
$cmd --version
$cmd --help
$cmd ([Aktion] | [Option])*  FILE ..

Aktionen
--tree                Datei DATADIR/tree.xml ($tree)
--kalender            Kalenderbilddateien ($kalender)
--xslt                XSLT-Datei erzeugen ($xslt)
--mkhelp              Hilfedatei ($mkhelp)
--base                Basisdaten ($base)
--langbase            Zuordnung der Sprachen zu Feiertags-Daten ($langbase)
--index               Index-Dateien ($index)
--htaccess            Datei KALDIR/b/.htaccess ($htaccess)

Optionen
--[no_]verbose        Erhöht den Umfang der Ausgabe des Scripts ($verbose)
--[no_]overwrite      Dateien ersetzen? ($overwrite)
--[no_]translate      Kalenderdateien übersetzen? ($translate)
--[no_]keeptmp        Temporäre Dateien behalten? ($keeptmp)
--basedir   BASEDIR   Basisverzeichnis zur Website
                      ($basedir)
--docroot   DOCROOT   Document Root des Servers
                      ($docroot)
--kaldir    KALDIR    Kalender-Unterverzeichnis der Document Root
                      ($kaldir)
--datadir   DATADIR   Verzeichnis der Kalender-Quelldaten
                      ($datadir)
--srcbase   SRCBASE   Basisverzeichnis der Skript-Dateien zur Website
                      ($srcbase)
--srcdir    SRCDIR    Verzeichnis der Skript-Dateien zum Kalender
                      ($srcdir)
--xsltpool  XSLTPOOL  Verzeichnis der gemeinsamen XSLT-Dateien
                      ($xsltpool)
--tempdir   TEMPDIR   Verzeichnis für temporäre Dateien
                      ($tempdir)
--trname    TRNAME    Kennung des Übersetzungsmoduls ($trname)
--trreplace TRREPLACE Ersetzung der Kennung der Übersetzers
                      ($trreplace)
--trlang    TRLANG    Zielsprachen ($trlang)
.HELP
} # show_help


# Zeigt die Version an
show_version ()
{
   cat << .VERSION ;
KLEIDER/web/src/kalender/kalender
Kalender auf der Website http://kleider.herbaer.de
2015-12-17 Herbert Schiemann <h.schiemann@herbaer.de>
GPL Version 2 oder neuer
.VERSION
} # show_version


# Variable und Zähler initialisieren
init_vars () {
   local v;
   declare_vars ;
   for v in $g_counters $g_variables $g_actions; do
      eval "$v=" ;
   done;
} # init_vars
```

```
# Argumente verarbeiten
read_args ()
{
   local wd ;
   local lastwd ;
   local var ;
   local ok ;

   has_actions=0 ;
   for wd in "$@"; do
      if [[ -n "$lastwd" ]]; then
         if [[ "$wd" =~ ^[\ a-zA-Z0-9./_#-]+$ ]]; then
            ok=0 ;
            for var in $g_variables; do
               if [[ "$var" == "$lastwd" ]]; then
                  (( ++ok )) ;
                  eval "$var=\"$wd\"" ;
                  break ;
               fi ;
            done ;
            if (( ! ok )); then
               (( verbose )) && echo "Unbekannte Option --$lastwd $wd" ;
               exit 11 ;
            fi ;
         else
            (( verbose )) && echo "Ungültiger Optionswert --$lastwd $wd" ;
            exit 12;
         fi;
         lastwd= ;
      else
         case "$wd" in
            --version )
               show_version ;
               exit 0 ;
               ;;
            --help )
               show_version ;
               show_help ;
               exit 0 ;
               ;;
            -- )
               if [[ -n "$_argv" ]]; then
                  lastwd=--;
                  continue;
               else
                  (( verbose )) && echo "Ungültige Option $wd" ;
                  exit 13 ;
               fi ;
               ;;
            --* )
               if [[ "$wd" =~ ^--[a-z][a-z0-9_]*$ ]]; then
                  lastwd=${wd#--} ;
                  ok=0 ;
                  for var in $g_counters ; do
                     if   [[ "$lastwd" == $var    ]]; then
                        eval "(( ++$lastwd ))" ;
                     elif [[ "$lastwd" == "no_$var" ]]; then
                        eval "${lastwd#no_}=0" ;
                     else
                        continue;
                     fi;
                     (( ++ok )) ;
                     break ;
                  done;
                  if (( !ok )); then
                     for var in $g_actions; do
                        if [[ "$lastwd" == "$var" ]]; then
                           eval "(( ++$var ))" ;
                           (( ++ok ));
                           has_actions=1;
                           break;
                        elif [[ "$lastwd" == "no_$var" ]]; then
                           eval "(( ++no_$var ))" ;
                           (( ++ok ));
                           break;
                        fi;
                     done;
                  fi;
                  (( ok )) && lastwd=;
               else
                  (( verbose )) && echo "Ungültige Option $wd" ;
                  exit 14 ;
               fi ;
               ;;
            * )
               if [[ -n $_argv ]]; then
                   _argv="$_argv $wd";
               else
                    (( verbose )) && echo "Ungültige Option $wd" ;
                    exit 15 ;
               fi;
               ;;
         esac ;
      fi ;
   done ;
   if [[ -n $lastwd && "$lastwd" != "--" ]]; then
```

```
      (( verbose )) && echo "Unverarbeitete Option --$lastwd";
      exit 16 ;
   fi ;
   [[ "$_argv" =~ ^[[:space:]]+$ ]] && _argv="" ;
} # read_args


# Aktionen ausführen
run_actions ()
{
   local act ;
   for act in $g_actions; do
     eval "(( ! has_actions && ! no_$act || $act )) && process_$act";
   done;
} # run_actions


# show_variables VARNAME1 VARNAME2
# Werte der Variablen anzeigen
show_variables ()
{
   local v ;
   for v in $g_counters $g_variables $g_actions $1; do
      eval "echo \"$v = \$$v\"" ;
   done;
} # show_variables


# ist ein Befehl verfügbar?
# check_command xsltproc sed
check_command ()
{
    local f ;
    for f in "$@"; do
        if [[ -z "$(which $f)" ]]; then
            (( verbose )) && echo "Befehl $f ist nicht verfügbar.";
            return 1;
        fi;
    done;
} # check_command


# Können die Eingabedateien gelesen werden?
# check_infiles first/path/to/file path/to/second_file ;
check_infiles ()
{
   local f ;
   for f in "$@"; do
      if [[ ! -f "$f" ]]; then
         (( verbose )) && echo "\"$f\" ist keine gewöhnliche Datei";
         return 1;
      fi;
      if [[ ! -s "$f" ]]; then
         (( verbose )) && echo "\"$f\" ist leer";
         return 1;
      fi;
      if [[ ! -r "$f" ]]; then
         (( verbose )) && echo "Kann Datei \"$f\" nicht lesen";
         return 1;
      fi;
   done;
   return 0;
} # check_infiles


# Können die Ausgabedateien erstellt werden?
# erstellt fehlende Verzeichnisse und löscht existierende Dateien
# nach Maßgabe der Variablen overwrite
# check_outfiles first/path/to/file path/to/second_file ;
check_outfiles ()
{
   local fp;
   local dir;
   local verb;
   (( verbose )) && verb=--verbose ;
   for fp in "$@"; do
      if [[ ! -e $fp ]]; then
         dir=${fp%/*};
         if [[ -n $dir && ! -e $dir ]]; then
            mkdir -p $verb $dir ;
            if [[ ! -d $dir ]]; then
               (( verbose )) && echo "$dir ist kein Verzeichnis";
               return 1;
            fi;
         fi;
      elif [[ -d $fp ]]; then
         (( verbose )) && echo "$fp ist ein Verzeichnis";
         return 1;
      elif (( overwrite )); then
         (( verbose )) && echo "lösche $fp";
         rm $fp;
      else
         (( verbose )) && echo "$fp existiert";
         return 1;
      fi;
      (( verbose )) && echo "$fp";
   done;
```

```
   return 0;
} # check_outfiles


# Sind die Dateien ausführbar?
# check_executeable first/path/to/script path/to/second_srcipt ;
check_executeable ()
{
   local f ;
   for f in "$@"; do
      if [[ ! -f "$f" ]]; then
         (( verbose )) && echo "$f\" ist keine gewöhnliche Datei";
         return 1;
      fi;
      if [[ ! -x "$f" ]]; then
         (( verbose )) && echo "$f\" ist keine ausführbare Datei";
         return 1;
      fi;
   done;
   return 0;
} # check_executeable


# gzip-komprimierte Datei(en) hinzufügen
add_gzip ()
{
   local f;
   for f in "$@"; do
      [[ -f $f ]] || continue;
      [[ -f $f.gz ]] && rm $f.gz;
      [[ -e $f.gz ]] && continue;
      (( verbose )) && echo "erstelle $f.gz";
      gzip --best --stdout $f > $f.gz ;
      (( verbose )) && echo "umbennen $f -> $f.";
      mv $f $f.;
   done ;
} # add_gzip


# Hilfsfunktion: temporäre Javascript und CSS-Dateien erzeugen
# proc_tempfiles subdir
proc_tempfiles ()
{
   local td=$1 ;
   local e ;  # Suffix js oder css
   for e in js css; do
      for s in $srcdir/*.$e ; do
         [[ -f $s ]] || continue;
         o=$td/${s#$srcdir/} ;
         check_outfiles $o && $srcbase/style/clean_$e.pl --in $s --out $o ;
      done;
   done;
} # proc_tempfiles


# Datei DATADIR/tree.xml
process_tree ()
{
   (( verbose )) && echo "process_tree" ;
   local o=$datadir/tree.xml ;
   check_outfiles $o || return ;
   $srcdir/tree.pl --kaldir "$datadir" --srcdir "$srcdir" --shiftnd none \
   | xsltproc $srcdir/tree_sorttlv.xslt - > $o                           ;
} # process_tree


# Hilfsfunktion: relativer Pfad zurück
# $(backpath auto/bahn/baustelle/datei) = ../../../
backpath ()
{
   local p=$1;
   local b=;
   local a;
   while [[ $p =~ ^([^/]*)/(.*)$ ]]; do
      a=${BASH_REMATCH[1]};
      p=${BASH_REMATCH[2]};
      [[ -n "$a"     ]] || continue;
      [[ "$a" == "." ]] && continue;
      b="../$b";
   done;
   echo $b;
} # backpath
```

```
# Kalenderdateien erstellen
process_kalender ()
{
   (( verbose )) && echo "process_kalender" ;
   local y    ; # relative URL einer Kalenderbild-Datei
   local q    ; # Kalenderbilder Quelldatei
   local r    ; # relativer Pfad der Kalenderbild-Datei
   local ss   ; # relativer Pfad zur XSLT-Datei
   local o    ; # Kalenderbilder-Zieldatei
   local t    ; # Kalenderbilder-Quelldatei mit Übersetzungen
   local b    ; # Übersetzungen zur Kalenderbilder-Quelldatei
   local td   ; # Unterverzeichnis für temporäre Dateien
   local t1   ; # Vortransformation zur Übersetzung
   local p    ; # Übersetzungsprogramm
   local t2   ; # Nachtransformation zur Übersetzung / Zusammenfügung
   local t3   ; # Transformation nach der Übersetzung
   local tr   ; # Dateien übersetzen?
   local pipe ; # Übersetzungs-Dienst starten?

   local $verb;
   (( verbose )) && verb=--verbose ;
   check_infiles                  \
     $srcdir/kal_addkeywords.xslt \
     $srcdir/add_sspi.xslt        \
     $xsltpool/xml_minimize.xslt  \
   || return                      ;
   td="$tempdir/$(date +%Y%m%d%H%M%S%N)" ;
   proc_shortids $td || return ;
   if (( translate )); then
      t1="$srcdir/lt_in.xslt" ;
      t2="$srcdir/lt_merge.xslt" ;
      t3="$srcdir/lt_chktrname.xslt" ;
      p="$srcdir/trans.pl" ;
      tr=1 ;
      check_infiles $t1 $t2 $t3 || tr=0 ;
      check_executeable $p      || tr=0 ;
      if (( tr )) && [[ $trname =~ "pipe" ]]; then
         if check_executeable                      \
            $srcbase/localization/pipe_srv.pl      \
            $srcbase/localization/pipe_srv_stop.pl  ;
         then
            pipe=1 ;
            $srcbase/localization/pipe_srv.pl $verb &
         else
            tr=0;
         fi;
      fi;
      (( tr )) || (( ! verbose )) || echo "Kalenderdateien werden nicht übersetzt." ;
   fi ;
   check_infiles $datadir/tree.xml || process_tree ;
   check_infiles $datadir/tree.xml || return 1 ;
   for y in $(xsltproc $srcdir/tree_files.xslt $datadir/tree.xml) ; do
      r=${y#kal/};
      q=$datadir/$r ;
      o=$kaldir/$r ;
      check_infiles $q || continue ;
      o=$kaldir/${q#$datadir/} ;
      check_outfiles $o $o. || continue ;
      if (( tr )); then
         t=$td/${q#$datadir/}
         b=$t.trans ;
         if check_outfiles $t $b; then
            xsltproc $t1 $q | $p --tgtlang "$trlang" --trname "$trname" > $b ;
            xsltproc --stringparam p_fnnew "$b" $t2 $q                \
            | xsltproc --stringparam p_replace "$trreplace" $t3 - > $t ;
         else
            t=$q ;
         fi ;
      else
         t=$q;
      fi;
      xsltproc                                     \
         --stringparam p_docroot  $docroot         \
         --stringparam p_shortids $td/shortids.xml \
         $srcdir/kal_addkeywords.xslt $t           \
      | xsltproc                                   \
         --stringparam p_ss $(backpath $r)s/kal.xslt \
         $srcdir/add_sspi.xslt -                   \
      | xsltproc $xsltpool/xml_minimize.xslt -     \
      > $o                                        ;
      add_gzip $o;
   done;
   (( pipe )) && $srcbase/localization/pipe_srv_stop.pl $verb ;
   (( keeptmp )) || rm --recursive $td ;
} # process_kalender
```

```
# Hilfsfunktion: Datei mit kurzen Text-Schlüsseln
# proc_shortids subdir
proc_shortids ()
{
   local sd=$1;
   (( verbose )) && echo "proc_shortids $1";
   local p=$srcbase/style/shortids.pl ;
   local t=$srcbase/style/localization_idlist.xslt ;
   local ids=$sd/shortids.xml ;
   local loc=$srcbase/style/local.xml.de ;
   check_executeable $p  || return 1 ;
   check_infiles $t $loc || return 1 ;
   check_outfiles $ids   || return 1 ;
   xsltproc $t $loc | $p > $ids;
   return 0 ;
} # proc_shortids


# XSLT-Datei erzeugen
process_xslt ()
{
   (( verbose )) && echo "process_xslt" ;
   local td ; # Unterverzeichnis für temporäre Dateien
   td="$tempdir/$(date +%Y%m%d%H%M%S%N)" ;
   local s ;  # XSLT-Quelldatei
   local n ;  # Basisname einer XSLT-Quelldatei
   local o ;  # Ausgabedatei
   proc_tempfiles $td;
   proc_shortids  $td  || return ;
   for n in kal treelist; do
      s=$srcdir/$n.xslt;
      [[ -f $s ]] || continue ;
      o=$kaldir/s/${s#$srcdir/} ;
      check_outfiles $o $o. || continue;
      xsltproc                                \
         --stringparam p_tmpprefix $td/       \
         --stringparam p_shortids $td/shortids.xml \
         $srcbase/style/styleincl_step_1.xslt $s   \
      | xsltproc --xinclude                   \
         $srcbase/style/styleincl_step_2.xslt -    \
      | $srcbase/style/rmxmlns.pl > $o        ;
      add_gzip $o ;
   done;
   (( keeptmp )) || rm --recursive $td ;
} # process_xslt


# Hilfe-Dateien erzeugen
process_mkhelp ()
{
   local h  ;  # Pfad einer Hilfe-Vorlage (PRESENTATION_help.xhtml.de)
   local b  ;  # Dateiname der Hilfedatei ohne Verzeichnispfad
   local tl ;  # Pfad der speziellen XSLT-Transformation
   local t  ;  # Pfad der XSLT-Datei help_step_1.xslt
   local t2 ;  # Pfad der XSLT-Datei help_step_2.xslt
   local tp ;  # Pfad der XSLT-Datei zur Ersetzung der Platzhalter
   local o  ;  # Pfad der Ausgabedatei
   local l  ;  # Kennung der Sprache
   (( verbose )) && echo "process_mkhelp" ;
   local td="$tempdir/$(date +%Y%m%d%H%M%S%N)" ;
   proc_tempfiles $td ;
   t=$srcbase/style/help_step_1.xslt ;
   t2=$srcbase/style/help_step_2.xslt ;
   tp=$srcbase/style/localization_repltext.xslt ;
   for h in $srcdir/*_help.xhtml.*; do
      [[ $h =~ ~$ ]] && continue ;
      b=${h#$srcdir/};
      o=$kaldir/s/$b;
      check_outfiles $o $o. || continue;
      l=${h##*.} ;
      tl=$src/${b%_help.xhtml.*}_mkhlp.xslt ;
      if check_infiles $tl; then
         xsltproc                                          \
            --stringparam p_local $srcbase/style/local.xml.$l \
            $tp $h                                         \
         | xsltproc $tl -                                  \
         | xsltproc $xsltpool/xhtml_minimize.xslt -        \
         | $srcbase/style/rmxmlns.pl > $o                  ;
      elif check_infiles $tp $t $t2; then
         xsltproc                                          \
            --stringparam p_local $srcbase/style/local.xml.$l \
            $tp $h                                         \
         | xsltproc --stringparam p_tmpprefix $td/ $t -    \
         | xsltproc --xinclude $t2 -                       \
         | $srcbase/style/rmxmlns.pl > $o                  ;
      fi;
      add_gzip $o ;
   done ;
   (( keeptmp )) || rm --recursive $td ;
} # process_mkhelp
```

```
# Basisdaten
process_base ()
{
   local i;   # Eingabedatei
   local o;   # Ausgabedatei
   local td ; # Unterverzeichnis für temporäre Dateien
   td="$tempdir/$(date +%Y%m%d%H%M%S%N)" ;
   (( verbose )) && echo "process_base" ;
   check_executeable $srcdir/base.pl || return;
   proc_shortids  $td  || return ;
   proc_base $td $datadir/feiertage;
   (( keeptmp )) || rm --recursive $td ;
} # process_base


# Hilfsfunktion: Unterverzeichnis oder Datei mit Basisdaten
proc_base ()
{
   local td=$1;        # Verzeichnis für temporäre Dateien
   local i=$2;         # Eingabedatei oder Verzeichnis
   local f;            # Verzeichniseintrag
   local o;            # Ausgabedatei
   local y;            # Jahr
   local ov=overwrite; # Sicherung overwrite
   if [[ $i =~ /(20[0-9][0-9])(/[A-Za-z./-]*)?$ ]]; then
      y=${BASH_REMATCH[1]};
   fi;
   if [[ -d $i ]]; then
      o=$i/a.xml ;
      o=$kaldir/b/${o#$datadir/feiertage/} ;
      if [[ -n "$y" ]] && check_outfiles $o $o.; then
         $srcdir/base.pl --year $y --f x --out $o. ;
         xsltproc                                          \
           --stringparam p_shortids $td/shortids.xml     \
           $srcbase/style/localization_shortids.xslt $o. \
         | xsltproc $xsltpool/xml_minimize.xslt -        \
         | $srcbase/style/rmxmlns.pl > $o                ;
         add_gzip $o;
      fi;
      for f in $i/*; do
         proc_base $td $f;
      done;
   else
      [[ $i =~ \.txt$ ]] || return ;
      o=${i%.txt}.xml;
      o=$kaldir/b/${o#$datadir/feiertage/} ;
      if [[ -z "$y" ]]; then
         y=$(date +%Y);
         f=$(date +%m);
         (( f > 10 )) && (( ++y ));
      fi;
      if check_infiles $i && check_outfiles $o $o.; then
         $srcdir/base.pl --year $y --ft $i --out $o. ;
         check_infiles $o. || return ;
         xsltproc                                          \
           --stringparam p_shortids $td/shortids.xml     \
           $srcbase/style/localization_shortids.xslt $o. \
         | xsltproc $xsltpool/xml_minimize.xslt -        \
         | $srcbase/style/rmxmlns.pl > $o                ;
         add_gzip $o;
      fi;
   fi;
}


# Hilfsfunktion: Grunddaten zu einer Sprache
proc_langbase ()
{
   local lang=$1;
   local ctr=$2;
   local d;
   local l;
   local o;
   local ok;
   if [[ -s $docroot/local/local.xml.$lang. ]]; then
      ok=0;
      for d in $kaldir/b/20[0-9][0-9]; do
         if [[ -s $d/${lang}-$ctr.xml. ]]; then
            l=$d/${lang}-$ctr.xml. ;
         elif [[ -s $d/$ctr.xml. ]]; then
            l=$d/$ctr.xml. ;
         else
            continue;
         fi;
         (( ++ok ));
         o=${l%/*}/a.xml.$lang.;
         check_outfiles $o || continue;
         ((verbose)) && echo "cp $l -> $o";
         cp $l $o;
         check_infiles ${l}gz || continue;
         check_outfiles ${o}gz || rm ${o}gz;
         cp ${l}gz ${o}gz;
      done;
      (( ok || ! verbose )) || echo "Keine Daten zur Sprache $lang / Land $ctr";
   else
      ((verbose)) && echo "Sprache $lang nicht angeboten";
```

```
   fi;
} # proc_langbase


# Basisdaten abhängig von der Sprache
process_langbase ()
{
   (( verbose )) && echo "process_langbase" ;
   check_infiles $srcdir/langcodes.dbd $srcdir/langcodes_cmd.xslt || return;
   eval $(xsltproc $srcdir/langcodes_cmd.xslt $srcdir/langcodes.dbd);
} # process_langbase


# Hilfsfunktion: eine Index-Datei
# proc_index ID REFBASE
proc_index ()
{
   local id=$1;
   local o=${2#kal};
   o=${o#/};
   local s=$datadir/$o; # Quellverzeichnis
   o=$kaldir/$o;        # Zielverzeichnis
   s=${s%/};
   o=${o%/};
   local q;             # Quelldatei
   local d;             # Zieldatei
   local n;             # Dateiname der Index-Datei

   d=$o/index.xhtml.de ;
   check_outfiles $d $d. ;
   for q in $s/index.xhtml.* ; do
      [[ $q =~ ~$ ]] && continue;
      check_infiles $q || continue;
      n=${q#$s/};
      n=${n%.}
      [[ $n =~ \.gz$ ]] && continue;
      d=$o/$n;
      check_outfiles $d $d. || continue;
      xsltproc $srcdir/xhtml_setlinks.xslt $q         \
      | xsltproc $srcdir/xhtml_settarget.xslt -       \
      | xsltproc $xsltpool/xhtml_minimize.xslt - > $d ;
      add_gzip $d;
   done;
   local ow=$overwrite;
   overwrite=0;
   d=$o/index.xhtml.de ;
   if check_outfiles $d $d. ; then
      local t=$srcdir/tree_ht.xslt ;
      local b=$datadir/tree.xml;
      check_infiles $t $b   || return ;
      xsltproc --stringparam p_id $id $t $b      \
      | xsltproc $xsltpool/xhtml_minimize.xslt - \
      | $srcbase/style/rmxmlns.pl                \
      > $d                                       ;
   fi;
   add_gzip $d ;
   overwrite=$ow ;
} # proc_index


# Index-Dateien
process_index ()
{
   (( verbose )) && echo "process_index" ;
   local t=$srcdir/tree_cmd.xslt ;
   local b=$datadir/tree.xml ;
   check_infiles $b    || process_tree ;
   check_infiles $t $b || return;
   eval $(xsltproc $t $b);
} # process_index


# DOCROOT/kal/b/.htaccess
process_htaccess ()
{
   (( verbose )) && echo "process_htaccess" ;
   local s=$srcdir/htaccess ;
   local o=$kaldir/b/.htaccess ;
   check_infiles $s && check_outfiles $o \
   && $srcbase/style/clean_config.pl --in "$s" --out "$o";
} # process_htaccess


# Sicherheit
export PATH=/bin:/usr/bin ;
IFS=$' \t\n' ;
init_vars ;
set -o noclobber ;   # existierende Dateien werden nicht überschrieben
shopt -s extglob nullglob ;
read_args "$@" ;
set_defaults ;
check_command xsltproc realpath date which || exit 1;
(( verbose > 1 )) && show_variables;
run_actions ;
exit 0;
```

# xhtml_settarget.xslt

[Quelltext]

# Namensräume

Die Namensraum-Präfixe, die aus dem erzeugten Dokument ausgeschlossen sind, sind durch einen Stern (*) in der ersten Spalte gekennzeichnet.

| | Präfix | Namensraum |
|---|---|---|
| | xml | http://www.w3.org/XML/1998/namespace |
| | (default) | http://www.w3.org/1999/xhtml |
| * | ht | http://www.w3.org/1999/xhtml |
| * | d | http://herbaer.de/xmlns/20051201/doc |
| | xsl | http://www.w3.org/1999/XSL/Transform |

# Parameter

## Parameter p_target

Default-Wert des Attributs `a/@target`

Select: '_blank'

Der Parameter wird in den folgenden Toplevel-Elementen benutzt:

Muster-Vorlage ht:a

# Muster-Vorlagen (matching templates)

## Muster-Vorlage /

Wurzel

## Muster-Vorlage ht:a

Verwendete globale Parameter oder Variable:

Parameter p_target

## Muster-Vorlage *

Alle anderen Elemente kopieren

## Muster-Vorlage @*

Attribute werden kopiert

# Quelltext

[Beschreibung]

```
<?xml version="1.0" encoding="utf-8"?>
<?xml-stylesheet href="xslt_ht.xslt" type="application/xml"?>
<!--
  Attribut a/@target setzen
-->
<xsl:stylesheet
  xmlns:xsl = "http://www.w3.org/1999/XSL/Transform"
  xmlns:d   = "http://herbaer.de/xmlns/20051201/doc"
  xmlns:ht  = "http://www.w3.org/1999/xhtml"
  xmlns     = "http://www.w3.org/1999/xhtml"
  exclude-result-prefixes = "d ht"
  version   = "1.0"
>
<xsl:param name = "p_target" select = "'_blank'"/>


<xsl:template match = "/">
  <xsl:apply-templates select = "comment() | processing-instruction() | *"/>
</xsl:template>


<xsl:template match = "ht:a">
  <xsl:copy>
    <xsl:apply-templates select = "@*"/>
    <xsl:if test = "not (@target)">
      <xsl:attribute name = "target">
        <xsl:value-of select = "$p_target"/>
      </xsl:attribute>
    </xsl:if>
    <xsl:apply-templates select = "* | text()"/>
  </xsl:copy>
</xsl:template>


<xsl:template match = "*">
  <xsl:copy>
    <xsl:apply-templates select = "@*|*|text()"/>
  </xsl:copy>
</xsl:template>


<xsl:template match = "@*">
  <xsl:copy-of select = "."/>
</xsl:template>


</xsl:stylesheet>
```

# Kalender für ein neues Jahr

Jedes Jahr muss ich mir die Erstellung der Kalender wieder in Erinnerung rufen. Deshalb stelle ich hier für mich die nötigen Schritte zusammen.

Zunächst ermittle ich die Daten der Feiertage in den verschiedenen Ländern (`kalender/feiertage/`*`JAHR/LANDKENNUNG`*`.txt`) und manchen Landesteilen (`kalender/feiertage/`*`JAHR/SPRACHE_LANDKENNUNG`*`.txt`).

Als nächstes erstelle ich eine Bildauswahldatei (`kalender/`*`JAHR/NAME`*`.xml`) sozusagen zum Üben. Ich kopiere die Datei `kalender/tags` als Vorlage nach `kalender/`*`JAHR/NAME`*`.xml` und bearbeite die Kopie. Wie kann ich jetzt den Kalender ansehen?

Ich erstelle ich die Grunddaten mit dem Befehl **`./kalender --base --langbase`** und einen symbolischen Verweis *`DOCROOT`*`/kal/`*`JAHR`* auf das Verzeichnis `kalender/`*`JAHR`*. Schon kann ich den Kalender unter der relativen URL `/kal/`*`JAHR/NAME`*`.xml` auf dem lokalen HTTP-Server ansehen.

Nach den Bildauswahldateien `kalender/`*`JAHR/NAME`*`.xml` erstelle ich die Index-Datei `kalender/`*`JAHR`*`/index.xhtml.de` für das Jahr *`JAHR`* und die neue Index-Datei `kalender/index.xhtml.de`.

Ich lösche die existierende Datei `tree.xml`, die Dateien *`DOCROOT`*`/kal/index.xhtml*` und den symbolischen Verweis *`DOCROOT`*`/kal/`*`JAHR`*. Der Befehl **`./kalender --tree --kalender --index`** (Skript `kalender`) erstellt die Dateien im Verzeichnis *`DOCROOT`*`/kal`.

Die Kalender müssen in den Haupt-Index aufgenommen werden. Ich lösche die Indexdateien *`DOCROOT`*`/index.xhtml*`. Der Befehl **`sitestyle/updweb --index`** erstellt die deutschsprachige Haupt-Indexdatei neu.

Es fehlen die Übersetzungen der Index-Dateien *`DOCROOT`*`/index.xhtml.de`, *`DOCROOT`*`/kal/index.xhtml.de` und *`DOCROOT`*`/kal/`*`JAHR`*`/index.xhtml.de`. Aber erst prüfe ich in Ruhe, ob lokal in der deutschsprachigen Version alles in Ordnung ist, und korrigiere vor der Übersetzung mögliche Fehler. Dann ist es Zeit, die neuen Dateien hochzuladen, und zwar zuerst die Grunddaten *`DOCROOT`*`/kal/b/`*`JAHR`*, dann die Kalender des kommenden Jahres *`DOCROOT`*`/kal/`*`JAHR`*, dann die Kalender-Indexdateien *`DOCROOT`*`/kal/index.xhtml.de.*`, schließlich die Haupt-Indexdatei *`DOCROOT`*`/index.xhtml.de.*`. Der Befehl **`localization/localize --overwrite --no_endaction index kal/index kal/`***`JAHR`***`/index`** erstellt die Übersetzungen und lädt sie hoch.

Jetzt sollte ich im „Begrüßungstext” der Website auf die neuen Kalender hinweisen. Ich bearbeite die Datei `components/news/index.dbk` und lösche die Dateien *`DOCROOT`*`/comp/news/index.xhtml*`. Der Befehl **`sitestyle/updweb --comp`** aktualisiert die lokale und die „richtige” Website einschließlich der Übersetzungen.

Schließlich sollte auch die Sitemap aktualisiert werden. Das erledigt der Befehl **`addstory/addstory --sitemapni --sitemap --uploadsm`**.

# monidfix.pl: IDs der Monatsnamen korrigieren

## Der Fehler

Die Kennungen der sprachabhängigen Texte in den Dateien *DOCROOT*/local/local.xml.*LANG*. sind geändert. Zu den sprachabhängigen Texten gehören die Monatsnamen. In den Kalendergrunddaten-Dateien *DOCROOT*/kal/b/*YEAR*/*COUNTRY*.xml. und *DOCROOT*/kal/b/*YEAR*/a.xml.*LANG*. stehen aber noch die alten Kennungen. Sie müssen durch die neuen Kennungen ersetzt werden. Wegen dieses Fehlers fehlen die Monatsnamen in den Kalendern.

Dazu benutze ich ein Perl-Skript.

## Die alten Kennungen

Die alten Kennungen werden in jeder Grunddaten-Datei benutzt. Ich lese sie aus der Datei *DOCROOT*/kal/b/2020/de.xml.. Die Platzhalter sind in der Form <l:ph id="*ID*"/> in der Datei enthalten. Die Datei enthält die Kalenderdaten nicht nur für das aktuelle Jahr, sondern auch für den Dezember des Vorjahres und den Januar des Folgejahres. Die Daten für jeden Monat enthalten den Platzhalter für den Monatsnamen meist mehrfach. Der folgende Perl-Code liest die alten Kennungen.

```
my $oldids = [];
my $fn   = "DOCROOT/kal/b/2020/de.xml.";
my $verb = 1;
my $h;
if (!open ($h, "<", $fn)) {
   print STDERR "Kann Datei \"$fn\" nicht öffnen\n";
   return;
}
my $d;
{
   local $INPUT_RECORD_SEPARATOR;
   $d = <$h>;
   close $h;
}
my $pid = "";
my $id  = "";
while ( $d =~ /<l:ph id="([a-z0-9]+)"\/>/g ) {
   $id = $1;
   if ($id ne $pid) {
      push (@$oldids, $id);
      $pid = $id;
   }
}
if ($verb) {
   $pid = 0;
   for $id (@$oldids) {
      print "$pid $id\n";
      ++$pid;
   }
}
```

## Die neuen Kennungen

Die Texte zu den Kennungen lese ich aus der Datei *DOCROOT*/local/local.xml.de. mit den deutschen Texten. Sie sind in der Form <t id="*ID*">*TEXT*</t> enthalten. Ich speichere die Kennungen zu allen Texten als $newids -> {"*TEXT*"} = "*ID*".

```
my $newids = {};
$fn = "DOCROOT/local/local.xml.de.";
$h  = undef;
if (!open ($h, "<:encoding(utf-8)", $fn)) {
   print STDERR "Kann Datei \"$fn\" nicht öffnen\n";
   return;
}
{
   local $INPUT_RECORD_SEPARATOR;
   $d = <$h>;
   close $h;
}
# <t id="cr">Januar</t>
while ( $d =~ /<t id="([a-z0-9]+)">([^>]+)<\/t>/g ) {
```

```
   $newids -> {$2} = $1;
}
my $names = [
   "Januar",
   "Februar",
   "März",
   "April",
   "Mai",
   "Juni",
   "Juli",
   "August",
   "September",
   "Oktober",
   "November",
   "Dezember",
],
if ($verb) {
   my $nm;
   my $i = 0;
   for $nm (@$names) {
      ++$i;
      print "$i $nm ", $newids -> {$nm}, "\n";
   }
}
```

# Alte und neue Kennungen zuordnen

Nun ordne ich die neuen Kennungen den alten Kennungen zu in der Form `$idmap -> {"OLD_ID"} = "NEW_ID"}`.

Der erste Eintrag in der Liste der alten Kennungen ist die Kennung für den Namen des Monats Dezember des Vorjahres. Ich ignoriere diesen Eintrag. Die nächsten zwölf Listeneinträge sind die Kennungen der Namen der Monate von Januar bis Dezember. Die Zuordnung ist einfach:

```
my $idmap = {};
shift @$oldids;
my $nm;
while (@$names) {
   $nm = shift @$names;
   $id = shift @$oldids;
   $idmap -> {$id} = $newids -> {$nm};
}
```

# Alte Kennungen ersetzen

Ich erstelle jetzt eine Funktion, die in einer Kalenderdaten-Datei die alten Kennungen durch die neuen Kennungen ersetzt. Ein regulärer Ausdruck findet alle Platzhalter der Form `<l:ph id="ID"/>`. Die alte `ID` ist die erste Match-Gruppe `$1` des regulären Ausdrucks. Eine Hilfsfunktion liefert den Platzhalter mit der neuen ID: `<l:ph id="NEWID"/>`. Eine alte Kennung, der nicht eine neue Kennung zugeordnet ist (also kein Monatsname), bleibt stehen.

```
my $subst = sub {
   $id = $idmap -> {$1} || $1;
   return "<l:ph id=\"$id\"/>";
};
```

Die Funktion zur Verarbeitung einer Datei nimmt zwei Parameter: `$in` ist der Dateipfad der Eingabedatei mit den alten Kennungen, `$out` ist der Dateipfad der Ausgabedatei mit den neuen Kennungen.

```
my $proc_file = sub {
   my ($in, $out) = @_;
   print "$in -> $out\n" if $verb;
   $h = undef;
   if (!open ($h, "<:encoding(utf-8)", $in)) {
      print STDERR "Kann Datei \"$in\" nicht öffnen\n";
      return;
   }
   {
      local $INPUT_RECORD_SEPARATOR;
      $d = <$h>;
      close $h;
   }
   $d =~ s/<l:ph id="([a-z0-9]+)"\/>/$subst -> ($idmap, $1)/ge ;
   $h = undef;
   if (!open ($h, ">:encoding(utf-8)", $out)) {
      print STDERR "Kann Ausgabedatei \"$out\" nicht öffnen\n";
```

```
         return;
      }
      print $h $d;
      close $h;
   };
```

# Alle Dateien verarbeiten

Alle Grunddaten-Dateien *DOCROOT*/kal/b/*YEAR*/*COUNTRY*.xml. und *DOCROOT*/kal/b/*YEAR*/a.xml.*LANG*. sind zu korrigieren. Für die korrigierten Dateien wähle ich statt *DOCROOT*/kal/b ein anderes Pfad-Präfix (*DOCROOT*/kal/bnew. Ich lese das Verzeichnis *DOCROOT*/kal/b und die Unterverzeichnisse *YEAR* und verarbeite die Grunddaten-Dateien:

```
   use File::Spec::Functions qw(catdir catfile);
   use File::Path qw(make_path);

   my $indir  = "DOCROOT/kal/b";
   my $outdir = "DOCROOT/kal/bnew";
   my ($dh, $sdh);
   my ($de, $sde);
   my $dp;
   my $od;
   opendir ($dh, $indir);
   while (defined ($de = readdir ($dh))) {
      next unless $de =~ /^20\d\d$/;
      $dp = catdir ($indir, $de);
      next unless -d $dp;
      $sdh = undef;
      opendir ($sdh, $dp)
      $od = catdir ($outdir, $de);
      make_path ($od);
      while (defined ($sde = readdir ($sdh))) {
         next unless $sde =~ /\.xml(?:\..+)?\.$/;
         $proc_file -> (catfile ($dp, $sde), catfile ($od, $sde));
      }
      closedir $sdh;
   }
   closedir $dh;
```

# Gzip-komprimierte Dateien

Zu jeder Grunddaten-Datei füge ich eine gzip-komprimierte Datei hinzu mit dem Suffix .gz. Dazu benutze ich ein bash-Skript:

```
#!/bin/bash
b=$(realpath $0);
b=${b%/src/*};
dir=$b/docroot/kal/bnew;
for sd in $dir/*; do
   echo "subdir $sd";
   f=${sd#$dir/};
   [[ $f =~ ^20[0-9][0-9]$ ]] || continue;
   [[ -d $sd ]] || continue;
   for f in $sd/*xml*\. ; do
      echo $f;
      gzip --best --stdout $f > ${f}gz ;
   done;
done;
```

# Die Skripte

Die vollständigen Skripte sind monidfix.pl und monidfix_addgz.

# monidfix.pl

[Quelltext]

## Übersicht

`monidfix.pl --help|--version`

`monidfix.pl[ --verbose …|--no_verbose ][ --docroot` *`DOCROOT`* `]`
`[ --oldidsrc` *`OLDIDSRC`* `][ --newidsrc` *`NEWIDSRC`* `][ --indir` *`INDIR`* `][ --outdir` *`OUTDIR`* `]`

## Beschreibung

Ich änderte einmal die Kennungen der Monatsnamen in den Lokalisierungsdateien Auch später kann ich vielleicht die Kennungen ändern. Dieses Skript ersetzt die alten Kennungen der Monatsnamen in den Kalendergrunddaten-Dateien durch die neuen Kennungen. Die angepassten Dateien werden in einem neuen Verzeichnis erstellt.

Es ist mit Perl 5.24.1 getestet.

Das Bash-Skript `monidfix_addgz` fügt gzip-komprimierte Versionen hinzu.

## Optionen

Zu allen Befehlszeilenargumenten gibt es Voreinstellungen.

`--help`

Gibt eine kurze Hilfe mit den Voreinstellungen zu allen möglichen Befehlszeilenargumenten aus.

`--version`

Gibt kurze Hinweise zum Programm und die Version aus.

`--verbose`

Erhöht den Umfang der Meldungen nach STDERR.

`--no_verbose`

Unterdrückt die Ausgabe von Meldungen. Die Optionen `--verbose` und `--no_verbose` werden der Reihe nach ausgewertet.

`--docroot` *`DOCROOT`*

Das `DOCUMENT_ROOT`-Verzeichnis des lokalen Webservers. In der Voreistellung sind die übrigen Dateipfad-Argumente Unterpfade dieses Verzeichnisses.

`--oldidsrc` *`OLDIDSRC`*

Der Dateipfad der Kalendergrunddaten-Datei, der die alten (zu ersetzenden) Kennungen der Monatsnamen entnommen werden.

`--newidsrc` *`NEWIDSRC`*

Der Dateipfad der Lokalisierungsdatei, der die neuen (einzusetzenden) Kennungen der Monatsnamen entnommen werden.

`--indir` *`INDIR`*

Der Pfad des Verzeichnisses mit den Kalendergrunddaten. Die Kalendergrunddaten-Dateien liegen in Unterverzeichnissen *`INDIR/JAHR`*. Ihr Dateiname enthält `.xml.` und endet mit einem Punkt (`.`).

`--outdir` *`OUTDIR`*

Die korrigierten Kalendergrunddaten-Dateien werden zur Sicherheit in einem anderen Verzeichnis *`OUTDIR`* erstellt.

# Benutzte Module

Das Programm benutzt außer den Standard-Modulen `File::Spec::Functions` und `File::Path` die folgenden Module:

`Herbaer::Readargs` (`Readargs.pm`)

Die Funktion `Herbaer::Readargs::read_args` liest die Befehlszeilen-Argumente.

`Herbaer::Replace`

Die Funktion `Herbaer::Replace::replace` ersetzt Platzhalter der Form `${xxx}`. Die Datei `Replace.pm` ist im Zusammenhang mit Lokalisierungen / Übersetzungen beschrieben.

# Quelltext

[Beschreibung]

```
#!/usr/bin/perl -w
# Kennungen der Monatsnamen in den Kalendern korrigieren
# 2020-09-30 Herbert Schiemann <h.schiemann@herbaer.de>


use utf8;                     # Dieser Quelltext ist utf-8-kodiert
use Cwd qw(realpath);
use English;
use File::Spec::Functions qw(catdir catfile);
use File::Path qw(make_path);
use Herbaer::Readargs;
use Herbaer::Replace;


binmode (STDIN,  ":encoding(utf-8)");
binmode (STDOUT, ":encoding(utf-8)");
binmode (STDERR, ":encoding(utf-8)");


my $args = {
   "[cnt]verbose" => 1,
   "docroot"      => undef,                              # lokale DOCUMENT_ROOT
   "oldidsrc"     => "\${docroot}/kal/b/2020/de.xml.",   # Quelle der alten IDs
   "newidsrc"     => "\${docroot}/local/local.xml.de.",  # Quelle der neuen IDs
   "indir"        => "\${docroot}/kal/b",                # Eingabeverzeichnis
   "outdir"       => "\${docroot}/kal/bnew"              # Ausgabeverzeichnis
};


# gibt die Version nach STDOUT aus
sub version {
   print << 'VERSION';
monidfix.pl
Kennungen der Monatsnamen in den Kalendern korrigieren
2020-09-30 Herbert Schiemann <h.schiemann@herbaer.de>
GPL 2 oder neuer
VERSION
};
$args -> {"[sr]version"} = sub { version (); exit 0; };


$args -> {"[sr]help"} = sub {
   set_defaults ($args);
   version ();
   print_message_with_values (<<"HELP", $args);
$0 --help     zeigt diese Hilfe an
$0 --version  zeigt die Programm-Version an

$0 [option]...
--[no_]verbose        erhöht den Umfang der STDERR-Ausgabe \${[cnt]verbose}
--docroot  DOCROOT    Pfad der lokalen DOCUMENT_ROOT
                      \${docroot}
--oldidsrc OLDIDSRC   Quelle der alten IDs mit Platzhalter <docroot>
                      \${oldidsrc}
--indir    INDIR      Eingabeverzeichnis
                      \${indir}
--outdir   OUTDIR     Ausgabeverzeichnis
                      \${outdir}
HELP
   exit 0;
}; # help


sub set_defaults {
   my $args = shift;
   my $b = realpath ($0);
   $b =~ s/\/src\/[^\/]*\/monidfix.pl//;
   $args -> {"docroot"}  ||= "$b/docroot";
   replace ($args, $args);
}; # set_defaults


my $data = {
   "names"  => [
      "Januar",
      "Februar",
      "März",
      "April",
      "Mai",
      "Juni",
      "Juli",
      "August",
      "September",
      "Oktober",
      "November",
      "Dezember",
      ],
   "oldids" => [],  # Liste der alten Kennungen 0 - 13
   "newids" => [],  # neue Kennungen
};
```

```
sub get_oldids {
   my ($args, $data) = @_;
   my $oldids = [];
   $data -> {"oldids"} = $oldids;
   my $fn   = $args -> {"oldidsrc"};
   my $verb = $args -> {"[cnt]verbose"};
   my $h;
   my $d;
   if (!open ($h, "<:encoding(utf-8)", $fn)) {
      print STDERR "Kann Datei \"$fn\" nicht öffnen\n";
      return;
   }
   {
      local $INPUT_RECORD_SEPARATOR;
      $d = <$h>;
      close $h;
   }
   my $pid = "";
   my $id  = "";
   # <l:ph id="en"/>
   while ( $d =~ /<l:ph id="([a-z0-9]+)"\/>/g ) {
      $id = $1;
      if ($id ne $pid) {
         push (@$oldids, $id);
         $pid = $id;
      }
   }
   if ($verb) {
      $pid = 0;
      for $id (@$oldids) {
         print "$pid $id\n";
         ++$pid;
      }
   }
}; # get_oldids


sub get_newids {
   my ($args, $data) = @_;
   my $newids = {};
   $data -> {"newids"} = $newids;
   my $fn   = $args -> {"newidsrc"};
   my $verb = $args -> {"[cnt]verbose"};
   my $h;
   my $d;
   if (!open ($h, "<:encoding(utf-8)", $fn)) {
      print STDERR "Kann Datei \"$fn\" nicht öffnen\n";
      return;
   }
   {
      local $INPUT_RECORD_SEPARATOR;
      $d = <$h>;
      close $h;
   }
   # <t id="cr">Januar</t>
   while ( $d =~ /<t id="([a-z0-9]+)">([^>]+)<\/t>/g ) {
      $newids -> {$2} = $1;
   }
   my $names = $data -> {"names"};
   if ($verb) {
      my $nm;
      my $i = 0;
      for $nm (@$names) {
         ++$i;
         print "$i $nm ", $newids -> {$nm}, "\n";
      }
   }
}; # get_newids


sub map_ids {
   my ($args, $data) = @_;
   my $oldids = $data -> {"oldids"};
   my $newids = $data -> {"newids"};
   my $idmap = {};
   $data -> {"idmap"} = $idmap;
   my $verb = $args -> {"[cnt]verbose"};
   my $names = $data -> {"names"};
   shift @$oldids;
   my $nm;
   my $id;
   while (@$names) {
      $nm = shift @$names;
      $id = shift @$oldids;
      $idmap -> {$id} = $newids -> {$nm};
   }
}; # map_ids
```

```
sub process_files {
   my ($args, $data) = @_;
   my $idmap = $data -> {"idmap"};
   my $verb  = $args -> {"[cnt]verbose"};
   my $id;
   my $subst = sub {
      $id = $idmap -> {$1} || $1;
      return "<l:ph id=\"$id\"/>";
   };
   my ($in, $out);
   my $h;
   my $d;
   my $proc_file = sub {
      ($in, $out) = @_;
      print "$in -> $out\n" if $verb;
      $h = undef;
      if (!open ($h, "<:encoding(utf-8)", $in)) {
         print STDERR "Kann Datei \"$in\" nicht öffnen\n";
         return;
      }
      {
         local $INPUT_RECORD_SEPARATOR;
         $d = <$h>;
         close $h;
      }
      $d =~ s/<l:ph id="([a-z0-9]+)"\/>/$subst -> ($idmap, $1)/ge ;
      $h = undef;
      if (!open ($h, ">:encoding(utf-8)", $out)) {
         print STDERR "Kann Ausgabedatei \"$out\" nicht öffnen\n";
         return;
      }
      print $h $d;
      close $h;
   };
   my $indir  = $args -> {"indir"};
   my $outdir = $args -> {"outdir"};
   my ($dh, $sdh);
   my ($de, $sde);
   my $dp;
   my $od;
   if (!opendir ($dh, $indir)) {
      print STDERR "Kann Verzeichnis \"$indir\" nicht lesen\n";
      return;
   }
   while (defined ($de = readdir ($dh))) {
      next unless $de =~ /^20\d\d$/;
      $dp = catdir ($indir, $de);
      next unless -d $dp;
      $sdh = undef;
      if (!opendir ($sdh, $dp)) {
         print STDERR "Kann Verzeichnis \"$dp\" nicht lesen\n";
         next;
      }
      $od = catdir ($outdir, $de);
      make_path ($od);
      while (defined ($sde = readdir ($sdh))) {
         next unless $sde =~ /\.xml(?:\..+)?\.$/;
         $proc_file -> (catfile ($dp, $sde), catfile ($od, $sde));
      }
      closedir $sdh;
   }
   closedir $dh;
}


read_args     ($args);
set_defaults  ($args);
get_oldids    ($args, $data);
get_newids    ($args, $data);
map_ids       ($args, $data);
process_files ($args, $data);
```

# Datei monidfix_addgz

```
#!/bin/bash
# -*- coding:utf-8 -*-
# file KLEIDER/web/src/kalender/monidfix_addgz
# 2020-09-30 Herbert Schiemann <h.schiemann@herbaer.de>
# gzip-komprimierte Dateien zu Kalenderdaten


b=$(realpath $0);
b=${b%/src/*};
dir=$b/docroot/kal/bnew;
for sd in $dir/*; do
   echo "subdir $sd";
   f=${sd#$dir/};
   [[ $f =~ ^20[0-9][0-9]$ ]] || continue;
   [[ -d $sd ]] || continue;
   for f in $sd/*xml*\. ; do
      echo $f;
      gzip --best --stdout $f > ${f}gz ;
   done;
done;
```