

Re: UTF8: cgi ist staerker als ich

Source: <http://newsgroups.derkeiler.com/Archive/De/de.comp.lang.perl.cgi/2007-10/msg00009.html>

- *From:* "Peter J. Holzer" <hjp-usenet2@xxxxxx>
 - *Date:* Sat, 20 Oct 2007 16:14:45 +0200
-

On 2007-10-17 08:43, Martin Trautmann <t-use@xxxxxxx> wrote:

On Tue, 16 Oct 2007 20:32:28 +0200, Helmut Wollmersdorfer wrote:

Das macht der Inline BIDI-Algorithmus des Browsers von selber – da hast Du keinen Einfluss darauf.

```
Ich verwende
use locale
use encoding "utf8"
```

Brauchst Du beides nicht.
use locale ist sogar äusserst gefährlich und unberechenbar.

Hm, ich sitze hier wohl in der Falle, dass die Umgebung latin1 ist, ich aber mit utf8 arbeiten will.

Nein. Du willst die Locale sowieso nicht verwenden, weil:

1) kannst Du sie gar nicht verwenden, weil Du ja – wie Du selber betont hast – Zeichen außerhalb des Repertoires von Latin-1 verwenden willst und somit das Repertoire Deiner Locale nicht ausreicht.

2) willst Du sie nicht verwenden, weil der Locale-Mechanismus Ende der 80er-Jahre für C entwickelt wurde und das damals herrschende Zeichensatz-Tohuwabohu widerspiegelt. 15 Jahre nach der Erfindung von Unicode und in Perl ist das alles sehr viel einfacher.

Also vergiss Locales und verwende den Unicode-Support von Perl.

Ich habe noch nicht wirklich verstanden, warum ein `encode_utf8` das UTF8-Flag wegnimmt.

Re: UTF8: cgi ist staerker als ich

Perl hat leider die unglückliche Eigenschaft, Implementierungsdetails in Namen offenzulegen. So heißen assoziative Arrays in Perl "Hashes" weil sie als Hashes implementiert sind. Sie könnten aber auch als Binäre Bäume, Tries, oder sonstwas implementiert sein, das würde für den Programmierer wenig Unterschied machen.

Bei "utf8" ist das ähnlich. Perl kodiert "wide character strings" intern als UTF-8. Solche Strings haben ein Flag gesetzt, das "UTF8-Flag" genannt wird, und sich mit der Funktion "is_utf8" abfragen lässt. Aber als Programmierer interessiert Dich nicht (und hat Dich auch nicht zu interessieren), dass das in UTF-8 kodiert ist. Perl könnte auch einfach 32 Bit pro Character verwenden, oder UTF-16 oder eine ander Kodierung. Alles was Dich interessiert, ist dass Du nicht 256 verschiedene Zeichen hast sondern ca. 2 (oder 4?) Milliarden und dass diese Zeichen als Unicode-Zeichen interpretiert werden, wo eine Interpretation notwendig ist ("`\x{FC}`") ist also immer ein "LATIN SMALL LETTER U WITH DIARESIS" und infolgedessen ein (Klein-)Buchstabe, völlig unabhängig von irgendwelchen Locales). Also vergiss, dass die irgendwas mit UTF-8 zu tun haben, nenne sie "Wide Character Strings" oder meinetwegen "Unicode-Strings" und betrachte den Namen `is_utf8` als historischen Unfall (so wie manche Programmiersprachen einen Funktion `ASC()` haben, die nur historisch was mit ASCII zu tun hat).

Wenn Dein Script mit der Außenwelt kommunizieren will (egal ob über Files, Pipes, Terminals oder ein Netzwerk), muss es das in Bytes tun. Es kann nur Bytes lesen und schreiben, und diese je nach Kontext vielleicht als Zeichen interpretieren. Perl kennt auch Byte-Strings, und diese haben das UTF8-Flag (das nicht so heißen sollte, siehe oben) NICHT gesetzt.

Beim encode kodierst Du Zeichen entsprechend einem bestimmten Encoding (z.B. Latin-1, UTF-8, CP850, UTF-16LE, EBCDIC, ...) als Bytes. Das Ergebnis ist ein Byte-String, daran erkennbar, dass das UTF8-Flag nicht gesetzt ist.

Beim decode dekodierst Du einen Bytestring in einem bestimmten Encoding zu Characters. Das Ergebnis ist ein (Wide) Character String, erkennbar am gesetzten UTF8-Flag.

Die Grundregel für das Verarbeiten von Texten in Perlscripts ist:

- * Alles was reinkommt, muss decodiert werden.
- * Alles was rausgeht, muss enkodiert werden.

In vielen Fällen gibt es Mechanismen, die das einigermaßen automatisch erledigen (IO-Layer, manche DBI-Module, offenbar auch aktuelle Versionen von CGI.pm), in anderen musst Du es explizit selber machen.

Re: UTF8: cgi ist staerker als ich

Die Ursache liegt derzeit wohl darin, dass das Perl-Script eigentlich in Latin-1 vorliegt.

Vielleicht. Was meinst Du überhaupt mit "Das Perl-Script liegt in Latin-1 vor"? Mit 'use encoding "utf8"' hast Du dem Compiler explizit gesagt, dass Dein Script (d.h. der Source-Code) in UTF-8 vorliegt. Wenn das nicht der Fall ist, kommt natürlich Unsinn heraus (eigentlich wundert mich, dass er es überhaupt kompilieren kann, aber man bekommt tatsächlich nur eine Warnung).

Fuer das CGI popup_menu muss ich die Umlaute wohl direkt angeben –

Was auch immer Du unter "direkt angeben" verstehst.

Ich prüfe aber auch mit is_utf8, ob der String wirklich utf-8 ist und nicht im Byte-Modus.

Und was machst du andernfalls?

decode oder upgrade aufrufen, vermutlich.

Und ich verwende nicht 'use CGI', sondern lese und schreibe mir das notwendige Zeugs (Header, Parameter) selber.

Das widerspricht meinem Verstaendnis von Modulen – sind die so schlecht, dass man sich nicht drauf verlassen kann?

Sturgeons Law trifft auch auf Perl-Module zu: 90 Prozent sind Schrott. CGI ist IMHO durchaus brauchbar, wenn es auch viele Dinge kann, die ich nie brauche, und dafür einige Dinge nicht bzw. nur umständlich erledigt, die ich sehr wohl brauche. Aber das ist halt so bei Modulen, die jemand anderer geschrieben hat – die decken die eigenen Bedürfnisse nie hundertprozentig ab.

Ich wuerde eher vermuten, dass der Fehler bei mir liegt.

Das würde ich auch vermuten.

hp

Re: UTF8: cgi ist staerker als ich

Re: UTF8: cgi ist staerker als ich

_ | Peter J. Holzer | I know I'd be respectful of a pirate
|_| | Sysadmin WSR | with an emu on his shoulder.
| | | hjp@xxxxxx |
_ / | <http://www.hjp.at/> | -- Sam in "Freefall"
.

Re: UTF8: cgi ist staerker als ich