Subject: Re: JavaScriptCore
Posted by mirek on Fri, 13 Aug 2010 07:09:04 GMT
View Forum Message <> Reply to Message

cbpporter wrote on Thu, 12 August 2010 03:55
After some superficial testing, running the current ToUpper/ToLower on the whole 65536 range I have found 568/569 errors. This is really great news, seeing as the table only covers 2048 characters. This means that most of Unicode is case agnostic and we can get away with good support without using huge tables. Running it on only 2048 characters, I have found 50/43. I hope I used the correct testing method.

So if somebody can point me to the bit setup of uni__info, I could correct the values for the first 2048 characters. I can figure out most, but it would be better if I could get the real layout of that packed bitfield. If there are any free bits left, I have some data that I would like to store there, like if the character is punctuation, if it is Latin, etc.

I would say this defines it pretty well:

```
bool IsLetter(int c)       { return (dword)c < 2048 ? uni__info[c] & 0xc0000000 : 0; }
bool IsUpper(int c)         { return (dword)c < 2048 ? uni__info[c] & 0x40000000 : 0; }
bool IsLower(int c)         { return (dword)c < 2048 ? uni__info[c] & 0x80000000 : 0; }
int  ToUpper(int c)        { return (dword)c < 2048 ? (uni__info[c] >> 11) & 2047 : c; }
int  ToLower(int c)        { return (dword)c < 2048 ? uni__info[c] & 2047 : c; }
int  ToAscii(int c)       { return (dword)c < 2048 ? (uni__info[c] >> 22) & 0x7f : 0; }
```

If bits are 0..31, then (reading the code, please check me...:):

31 lower letter
30 upper letter
22-28 (7 bits) toascii
11-21 (11 bits) toupper
0-10 (11 bits) tolower

Looks like bit 29 is now free, if I am counting well..

It would be nice to know, after fixing <2048, what are codepoints of those more errors - perhaps we could handle them too...