
Subject: Re: 16 bits wchar

Posted by [cbpporter](#) on Fri, 12 Oct 2007 09:27:33 GMT

[View Forum Message](#) <> [Reply to Message](#)

And the function for error-escaping:

```
inline byte * putUtf8(byte *s, int codePoint)
{
    if (codePoint < 0x80)
        *s++ = codePoint;
    else if (codePoint < 0x0800)
    {
        *s++ = 0xC2 | (codePoint >> 6);
        *s++ = 0x80 | (codePoint & 0x3f);
    }
    else if (codePoint < 0xFFFF)
    {
        *s++ = 0xE0 | (codePoint >> 12);
        *s++ = 0x80 | (codePoint >> 6) & 0x3F;
        *s++ = 0x80 | (codePoint & 0x3F);
    }
    else
    {
        *s++ = 0xF0 | (codePoint >> 18);
        *s++ = 0x80 | (codePoint >> 12) & 0x3F;
        *s++ = 0x80 | (codePoint >> 6) & 0x3F;
        *s++ = 0x80 | (codePoint & 0x3F);
    }
    return s;
}
```

```
String ToUtf8EE(const char *_s, int _len)
{
    int tlen = utf8lenEE(_s, _len);
    StringBuffer result(tlen);

    byte *s = (byte *) _s;
    const byte *lim = s + _len;

    byte *z = (byte *) ~result;
    int length = 0;
    while(s < lim) {
        int lmod, dep;
        int codePoint = utf8codepointEE(s, lim, lmod, dep);
        if (codePoint == -1)
            return "";
    }
```

```
length += lmod;
s += dep;

z = putUtf8(z, codePoint);
}
ASSERT(length == tlen);
return result;
}
```

Now I only need to implement the reverse operation and do some round-trip conversions (a large number of random chars should do the trick) to make sure everything is correct.
