

---

Subject: Re: C strings  
Posted by [mirek](#) on Wed, 11 Oct 2006 02:26:59 GMT  
[View Forum Message](#) <> [Reply to Message](#)

---

Quote:  
(or when it is negligible performance wise).

Quote:  
Not in all cases. For example,

```
void CodeEditor::Enclose(const char *c1, const char *c2)
{
    int l, h;
    if(!GetSelection(l, h))
        return;
    Insert(l, WString(c1));
    Insert(h + strlen(c1), WString(c2));
    ClearSelection();
    SetCursor(h + strlen(c1) + strlen(c2));
}
```

Calculating the size of const string is already overhead.

You are not seeing the whole story. Enclose is always called with const char \* literal parameters (of short length). So making parameters String would enlarge the code size, as strlen has to be done anyway and you would just have to call String constructor twice and before each Enclose call.

Quote:  
For non-const strings, determination of the end of string by zero terminator is unsafe, because it may be lost, and work with this string can make security hole.

I do not think this is true. How could that be lost?

The fixed buffers for strings are security reasons, not trailing '\0' sentinels.

Quote:  
I know and understand, that removing C-like strings is huge work (and code works without it, really? ), but IMHO, if you attempted on STL, you can make library safer and faster.

Actually, that would be nice, but is a bit problematic because of nature of C++ string literals and because of external environment. (C++ string literals being the major problem).

---