
Subject: [FIXED] - Memory leak on empty CtrlLib app on single PC

Posted by [mdelfede](#) on Fri, 05 Mar 2010 19:17:29 GMT

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

I've got one of my Ubuntu Karmik 64 machine that gives a weird memory leak on an empty basic CtrlLib app.

The weird stuff is that another machine with same compiler doesn't show the error.

After a regression testing I found a patch that triggers the error, svn revision 2141, file

Draw/Font.cpp :

```
void Font::InitStdFont()
{
    ONCELOCK {
        DrawLock __;
        List();
        AStdFont = Arial(12);
        String name;
        int height = 0;
        GetStdFontSys(name, height);
        int q = FindFaceNameIndex(name);
        if(q <= 0)
            q = FindFaceNameIndex("Tahoma");
        if(q <= 0)
            q = FindFaceNameIndex("Microsoft Sans Serif");
        if(q <= 0)
            q = FindFaceNameIndex("MS Sans Serif");
        if(q > 0) {
            AStdFont = Font(q, max(height, 1));
#ifdef PLATFORM_WIN32
            SyncStdFont();
#endif
        }
#ifdef PLATFORM_WIN32
        SyncStdFont();
#endif
    }
}
```

Taking off the calls of FindFaceNameIndex() for the 3 named fonts makes the bug disappear :

```
void Font::InitStdFont()
{
    ONCELOCK {
        DrawLock __;
        List();
```

```

AStdFont = Arial(12);
String name;
int height = 0;
GetStdFontSys(name, height);
int q = FindFaceNameIndex(name);
if(q > 0) {
    AStdFont = Font(q, max(height, 1));
#ifdef PLATFORM_WIN32
    SyncStdFont();
#endif
}
#ifdef PLATFORM_WIN32
    SyncStdFont();
#endif
}
}

```

I guess there's some problem in FindFaceNameIndex()....

Ciao

Max

edit : after a small check, the issue disappears also taking out the SyncStdFont() call. Maybe some un-released data from inside it ?

Max

edit #2 : I think it's a problem of unfreed FT_Face in FontFc cache system, but I could be wrong.

```

CommonFontInfo GetFontInfoSys(Font font)
{
    if(GetFontInfoSysXft)
        return (*GetFontInfoSysXft)(font);
    CommonFontInfo fi;
    String path;
    FT_Face face = FTFace(font, &path);
    if(face) {
        fi.ascent = face->size->metrics.ascender >> 6;
    }
}

```

Taking out the FTFace call (and setting face to NULL) the problem disappears too. I don't know enough in deep the font caching of upp, so I'll stop here the analysis

Max
