
Subject: Re: The very strange xterm issue
Posted by [mirek](#) on Mon, 17 Mar 2008 15:22:01 GMT
[View Forum Message](#) <> [Reply to Message](#)

In ide/Host.cpp:

```
void LocalHost::Launch(const char *_cmdline, bool console)
{
    String cmdline = FindCommand(exedirs, _cmdline);
    PutVerbose(cmdline);
#ifdef PLATFORM_WIN32
    if(console)
        cmdline = GetExeFilePath() + " ! " + cmdline;
    int n = cmdline.GetLength() + 1;
    Buffer<char> cmd(n);
    memcpy(cmd, cmdline, n);
    SECURITY_ATTRIBUTES sa;
    sa.nLength = sizeof(SECURITY_ATTRIBUTES);
    sa.lpSecurityDescriptor = NULL;
    sa.bInheritHandle = TRUE;
    PROCESS_INFORMATION pi;
    STARTUPINFO si;
    ZeroMemory(&si, sizeof(STARTUPINFO));
    si.cb = sizeof(STARTUPINFO);
    Buffer<char> env(environment.GetCount() + 1);
    memcpy(env, environment, environment.GetCount() + 1);
    if(CreateProcess(NULL, cmd, &sa, &sa, TRUE,
        NORMAL_PRIORITY_CLASS|CREATE_NEW_CONSOLE,
        ~env, NULL, &si, &pi)) {
        CloseHandle(pi.hProcess);
        CloseHandle(pi.hThread);
    }
    else
        PutConsole("Unable to launch " + String(_cmdline));
#endif
#ifdef PLATFORM_POSIX
    /* if(console) // why this does not work?!
        cmdline = "/usr/bin/xterm -hold -e " + cmdline;*/
    Buffer<char> cmd_buf(strlen(cmdline) + 1);
    char *cmd_out = cmd_buf;
    Vector<char *> args;
    const char *p = cmdline;
    const char *b = p;
    while(*p && (byte)*p > ' ')
        if(*p++ == '\\')
            while(*p && *p++ != '\\')
                ;
    ;
#endif
}
```

```

const char *app = cmd_out;
args.Add(cmd_out);
memcpy(cmd_out, b, p - b);
cmd_out += p - b;
*cmd_out++ = '\0';

while(*p)
if((byte)*p <= ' ')
    p++;
else {
    args.Add(cmd_out);
    b = p;
    while(*p && (byte)*p > ' ')
        if(*p++ == '\\')
        {
            memcpy(cmd_out, b, p - b - 1);
            cmd_out += p - b - 1;
            b = p;
            while(*p && *p != '\\')
                p++;
            memcpy(cmd_out, b, p - b);
            cmd_out += p - b;
            if(*p == '\\')
                p++;
            b = p;
        }
    memcpy(cmd_out, b, p - b);
    cmd_out += p - b;
    *cmd_out++ = '\0';
}

```

```
args.Add(NULL);
```

```

ONCELOCK {
    struct sigaction sigchld_action;
    memset(&sigchld_action, 0, sizeof(sigchld_action));
    sigchld_action.sa_handler = sCleanZombies;
    sigaction(SIGCHLD, &sigchld_action, NULL);
}

```

```

pid_t pid = fork();
if(pid == 0)
{
    const char *from = environment;
    Vector<const char *> env;
    while(*from)
    {
        env.Add(from);
    }
}

```

```
    from += strlen(from) + 1;
}
env.Add(NULL);
const char **envp = env.Begin();
execve(args[0], args, (char *const *)envp);
}
LLOG("Launch pid: " << pid);
sPid().Add(pid);
#endif
}
```

Mirek
