Using the same compiler as above, I had to change the sDumpWindow function in
Win32Wnd.cpp:

```
static BOOL CALLBACK sDumpWindow(HWND hwnd, LPARAM lParam) {
 String dump;
 dump << (IsWindowEnabled(hwnd) ? "ena" : "dis") << ' '
    << (IsWindowVisible(hwnd) ? "vis" : "hid") << ' '
    << Sprintf("owner=0x%x ", GetWindow(hwnd, GW_OWNER));
 Ctrl *ctrl = Ctrl::CtrlFromHWND(hwnd);
 if(ctrl) {
#ifdef _DEBUG
  dump << "Ctrl: " << UPP::Name(ctrl);
#endif
 }
 else if(!lParam)
  return TRUE;
 else
 {
#ifdef PLATFORM_WINCE
  wchar clsname[256], title[1024];
#else
  char clsname[256], title[1024];
#endif
  ::GetClassName(hwnd, clsname, __countof(clsname));
  ::GetWindowText(hwnd, title, __countof(title));

#ifdef PLATFORM_WINCE
 WString wClsName = clsname;
 WString wTitle = title;
  dump << "HWND: " << Sprintf("0x%x", hwnd) << ", class = "
    << wClsName << ", title = " << wTitle;
#else
  dump << "HWND: " << Sprintf("0x%x", hwnd) << ", class = "
    << clsname << ", title = " << title;
#endif
 }
 LLOG(dump);
 return TRUE;
}
```

I added the second #ifdef. The
I am surprised you didn't have problems with this - as apparently there is no AsString for wchar *
since the template version uses ToString, and wchar is a primitive type. There is also a
non-template version for PLATFORM_MSC that calls FormatPtr - for some reason I couldn't seem

to use this (since the compiler decides that the template version is closer for wchar *).

Now I have a small problem with the compiler itself - after that I think it will all compile correctly.