

Launch_S60_GUI_from_console

It is actually possible to bring up whole GUI from your console application.

There are several approaches.

Before justify what approach shall be taken to complete this unusual job, there are questions shall be answered first.

Question 1:

What are the controls you need? It makes a lot of differences between self drawing control and a compound control. Different controls might require different UI environments setup.

For example, a CEikTextListBox and CEikLabel needs not to link with avkon.lib, and CEikonEnv setup shall be sufficient in manipulating these two controls.

A CEikEdwin, on the other hand, shall be linked to avkon.lib, and a AvkonEnv UI setup will be needed.

Question 2:

Do you need to bring up UI once only or you need to do it more than one time?

Assuming your console application has its own task. e.g.: a listening socket server, you probably will need to create another thread for UI.

Case 1:

A GUI page shall be brought up on demand, by background running console program. The GUI contains all "simple" controls, i.e.: no need to link with avkon.lib.

Solution 1: create CEikonEnv using 2 phase construction in your E32Main()

```
GLDEF_C TInt E32Main()
{
    // WINS already creates environment for us
    CTrapCleanup* cleanup = CTrapCleanup::New();
    ::Delete("cleanup stack inited.");
    = new CEikonEnv;

    (error = TRAPD(e->ConstructL()));
    __ASSERT_ALWAYS(!error, User::Panic(_L("EXECTRL"), err));

    TRAPD(error, ExeMainL());
    __ASSERT_ALWAYS(!error, User::Panic(_L("EXECTRL"), error));

    //User::After(6*1000*1000);
    //User::After(3*1000*1000);

    //TODO: find out what happens when deleting coe.
    delete coe;
    delete cleanup;

    return 0;
}
```

Launch_S60_GUI_from_console

```
}

void __cdecl eMainL()
{
    coe->RootWin().SetOrdinalPosition(0, ECoeWinPriorityAlwaysAtFront);

    CMyControl* ctrl = CMyControl::NewL(TRect(10, 10, screenWidth, screenHeight));
    ctrl->DrawNow();

    for (;;)
    {
        // Wait synchronously for event
        TRequestStatus status;
        coe->WsSession().EventReady(&status);
        User::WaitForRequest(status);

        if (status.Int() == KErrNone)
        {
            TWsEvent event;
            coe->WsSession().GetEvent(event);

            // Check exit key
            if (event.Key()->iCode == EKeyDevice3)
                break;

            // Pass event to control
        }
    }
    if(ctrl)delete ctrl;
}
```

Solution 2:

Create Application, Document, AppUi, View like a standard GUI application, and use following line to invoke it

```
TRAPD(err, EikStart::RunApplication(NewApplication));
__ASSERT_ALWAYS(!err, User::Panic(_L("launching gui"), err));
```

Note, the EikStart::RunApplication call will block until the GUI application is exited, you might consider to do this in a new thread.

Case 1:

A GUI page shall be brought up on demand, by background running console program. The GUI contains some control like CEikEdwin that you need a full UI env to support it.

Solution:

same as solution 2 in case 1.