# 視窗環境應用程式設計期中考解答

姓名：_____

系級：_____

學號：_____

95/11/16 10:30-12:10

考試規則：1. 不得翻閱任何參考書籍或是資料
2. 不得使用任何形式的電腦 (包含計算機)
3. 不得左顧右盼、不得交談、不得交換任何資料、**試卷題目有任何疑問請舉手發問**
4. 如果不了解中文(英文)翻譯名詞或是說法的話，請提出來我會替你翻譯為英文(中文)，可以用英文或是中文回答
5. 提早繳卷同學請直接離開教室，不得逗留喧嘩
6. 違反上述任何一點之同學以作弊論，一律請校方處理

作答原則：1. 請在答案卷上註明題號作答，**如需回答在題目卷上，請在答案卷上註明**
2 請解釋你的答案，原因合理而答案略有出入是有部份分數的

1. (10%) 請問平常我們說什麼是 Win32 SDK 視窗應用程式? 什麼是 MFC 視窗應用程式?

**Sol:**

Win32 SDK 應用程式和 MFC 應用程式指的都是在微軟視窗系統下可以具有圖形化界面 (Graphic User Interface, GUI) 的應用程式，SDK 應用程式指的是直接基於微軟視窗系統提供出來的 C 函式庫 Win32 應用程式界面 (Windows 32 bit Application Interface, WIN32 API) 來製作的視窗應用程式，MFC 應用程式指的是運用 Microsoft Foundation Class, MFC 這一組 C++ 類別函式庫來撰寫視窗使用者界面的應用程式，MFC 類別函式庫基本上將 WIN32 API 包裝成一個一個 C++ 類別，同時也結合多個 WIN32 API 組合成一些程式設計樣式 (Design Pattern) 以及常用的使用者介面，以方便程式設計者使用。由於不需要處理很多 API 呼叫的細節，同時可以使用其預設的樣式及常用的使用者介面，所以程式設計者可以專注於更多應用功能的設計。

2. (5%) 請問為什麼具有視窗界面的應用程式比較適合使用訊息驅動的方式來撰寫呢? (用 C 裡頭函式呼叫的方式不行嗎?)

**Sol:**

最主要的原因是具有視窗介面的程式在運作時由“操作程式的人”來控制整個程式的運作，控制 CPU 執行程式的哪一部份，應用程式則是配合“操作程式的人”來執行需要的動作，而不是由應用程式本身來決定到底該執行哪一部份 (傳統程序式的程式是由程式邏輯來控制 CPU 的動作，操作程式的人只是被動地配合程式的邏輯來輸入資料而已)，這種分工執行的方式在程序式的程式運作中比較難實現，但是在物件化的程式運作中就只是“訊息 – 回應”的基本動作而已。這也就是為什麼微軟視窗在 C 的環境中設計了“訊息迴圈、訊息、和 callback 函式”等等機制的原因，藉由這樣的機制在程序式的程式環境中實現物件式的運作機制。

3. (10%) 在微軟系統中一個一般無視窗界面的執行檔案 (PE 檔. *.exe) 中，主要包含的是動態定址的資訊、偵錯用的表格、以及 CPU 可以執行的程式指令；而一個具有視窗界面的應用程式執行檔案中除了以上資訊外，為了方便使用者界面的設計，還包含所謂的“資源

(resource)”，請問所謂的資源包含了哪些東西? (Hint: 在透過 Visual Studio IDE 環境撰寫你的程式的時候你在 “資源檢視中可以看到哪些東西”)

**Sol:**

對話盒, 選單, 快速鍵, 字串, 工具列, 點陣圖, 圖示, 游標, 字型...

4. 請回答下列撰寫微軟視窗應用程式相關的問題:

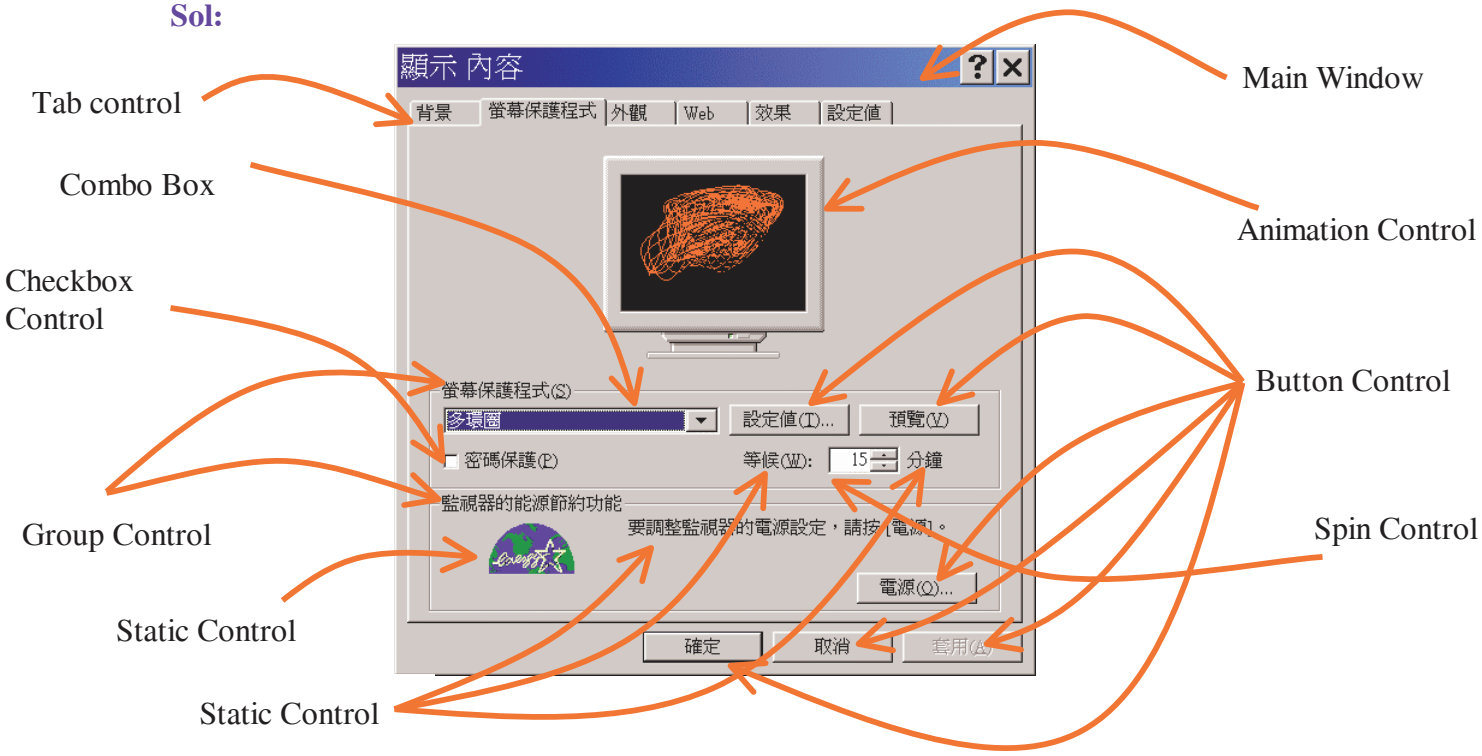
(a) (5%) 在撰寫 SDK 視窗應用程式時，我們用什麼 Win32 API 來開啟一個單一視窗?

**Sol:**

CreateWindow() 或是 CreateWindowEx()

(b) (10%) 對話盒視窗通常都是由很多單一視窗組合而成，在下圖對話盒視窗中，你認為可能有哪些視窗組合而成? (請標示在圖上)

**Sol:**




(c) (5%) 請問在撰寫 SDK 應用程式時我們用什麼 API 來開啟一個對話盒視窗? 需要配合什麼資源?

**Sol:**

DialogBox() 或是 CreateDialog()，需要傳給他對話盒配置的資源 ID，在這個 API 中會根據對話盒資源的配置資料來開啟指定的控制項視窗

(d) (5%) 請問 (c) 中所完成的工作能夠用 (a) 中的 API 來完成嗎? (如果可以, 請說明大概需要怎麼做)

**Sol:**

可以自己用 CreateWindow() API 來完成上述 DialogBox() 中所完成的事項，只要一個控制項一個控制項運用 CreateWindow() 在適當的位置上以視窗系統中預設的控制項視窗類別開啟子視窗即可

5. 請回答下列 SDK 應用程式與 MFC 應用程式撰寫時的相關問題：

(a) (5%) 請問 SDK 應用程式中的程式進入點 WinMain() 函式到了 MFC 應用程式中還存在嗎? 需要程式設計者提供嗎?

**Sol:**

還在，但是不需要由程式設計者提供，在動態連結的函式庫中，由於所需要執行的動作一成不變，所以 MFC 環境中是一個固定不需要修改的函式，它還是程式的進入點

(b) (5%) 請問 SDK 應用程式中的訊息迴圈 (Message Loop) 到了 MFC 應用程式中隱藏在哪一個類別的哪一個成員函式中?

**Sol:**

隱藏在 CWinThread::Run() 中

(c) (5%) 請問 SDK 應用程式中每一個視窗都有的視窗函式到了 MFC 應用程式中隱藏在什麼地方?

**Sol:**

CWnd 類別中有一個 static 的 Window procedure, 這個函式一般情況下我們也不需要去更改它,基本上它根據 Message Map 表格的內容來呼叫每一個訊息對應的訊息處理成員函式

(d) (5%) 請問 SDK 應用程式中視窗函式裡分配處理各種訊息的 switch 結構被 MFC 應用程式的什麼架構取代?

**Sol:**

Message Map 訊息對應表

6. (10%) 如右圖，在對話盒中有一個按鈕，請問使用者按下按鈕的時候會依序產生哪些訊息給哪些視窗? 有哪些視窗的視窗函式會被呼叫?


**Sol:**

右圖中主要有兩個視窗，主要的對話盒視窗和按鈕視窗，當使用者按下按鈕的時候：

1. 產生一個滑鼠左鍵按下及放開的訊息 WM_LBUTTONDOWN 及 WM_LBUTTONUP 給按鈕視窗，對應此兩個訊息，按鈕視窗的視窗函式被呼叫到
2. 按鈕視窗處理 WM_LBUTTONUP 的訊息，並且轉送一個 WM_COMMAND 的訊息給它的父視窗 (也就是主要的對話盒視窗)，表示子控制項視窗被按下了，此時外層對話盒視窗的視窗函式會被呼叫來處理這個訊息

7. (10%) 請問為什麼希望一個視窗的視窗函式中所有的輸出繪圖程式碼都集中放在處理 WM_PAINT 訊息的地方?

**Sol:**

在撰寫具有視窗介面的應用程式時，每一個視窗函式當然要自己負責繪製視窗的外觀 (包括所顯示的資料與圖形)，但是什麼時候該在視窗內畫圖，在視窗內的哪一區域內畫圖卻不見得是由應用程式來全權控制的，畫圖的時機主要有兩種：第一是程式流程中需要在視窗裡顯示一些資料，也就是程式邏輯中需要畫圖，例如一個移動的車子，每秒鐘顯示的位置和角度都不太一樣，第二種則是程式操作者運用滑鼠將視窗移動、最大化、

由別的視窗下面拉出等等不是由程式邏輯控制的顯示，應用程式應該要在最短的時間內去處理顯示的要求，否則螢幕上就顯示一塊空白的區域。

8. (10%) 我們說視窗系統在設計及運作時具有很清楚的物件概念，主要是在系統中具有很清楚的封裝概念和多型概念，在微軟視窗系統中請舉例說明你看到它如何實現封裝，如何實現多型? (Hint: 從你所知到的視窗基本運作方法對應到物件導向的概念來說明)

**Sol:**

**封裝:**

(1) 視窗物件

視窗系統將視窗物件維持在 USER32 模組內，程式設計者沒有辦法直接存取它的各種屬性，必須透過介面函式，例如 SetWindowText(), SetWindowPos() 等等，另外每一個包裝好的視窗物件也透過視窗函式 (window procedure) 來描述它的行為，這都是良好的封裝與抽象化

(2) GDI 物件

(3) DeviceContext 物件

…

**多型:**

(1) 視窗物件 (按鈕視窗, 對話盒視窗, 輸入欄位視窗, Checkbox, Radio button, …)

各種視窗物件雖然有不同的屬性和不同的行為表現，但是對於視窗系統來說，在訊息處理的基本機制，視窗開啟關閉，視窗放大縮小，視窗移動等等方面都是一致的

(2) 顯示界面 (螢幕, 印表機)

對於視窗應用程式來說，螢幕和印表機都是一致的 GDI (Graphic Device Independent) 輸出界面，具有 Device Context 描述其特性

…