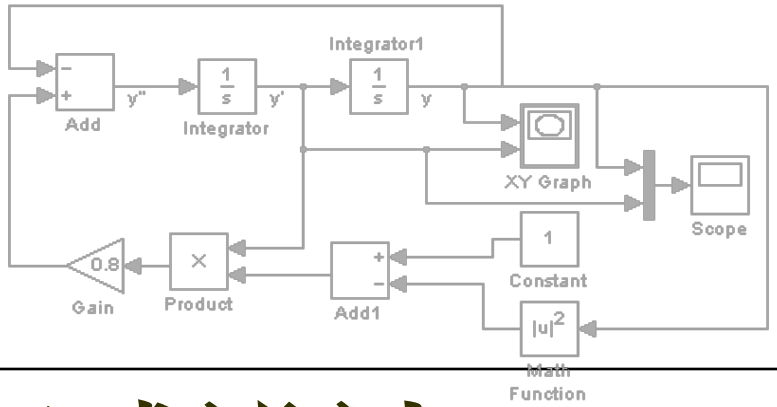


第十八章

進階GUI程式設計

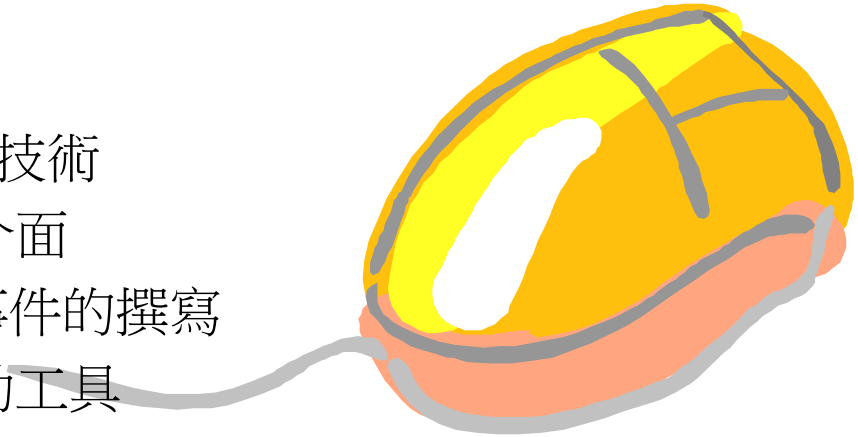
本章學習目標

學習switch-yard的撰寫技術

利用GUIDE設計視窗介面

學習功能表的設計與事件的撰寫

學習GUIDE其它的輔助工具



18.1 利用switch-yard技術撰寫GUI

18.1.1 switch-yard技術的基本認識

- switch-yard 技術：

  利用switch 來判別是哪一段程式碼該被執行

  要取得某個元件的handle，可用如下的語法來設定：

```
set(h,'Tag','tag_name');
```

  要找尋某個元件的handle時，可以利用findobj函數：

```
h=findobj(0,'Tag','tag_name');
```

18.1.2 簡單的範例

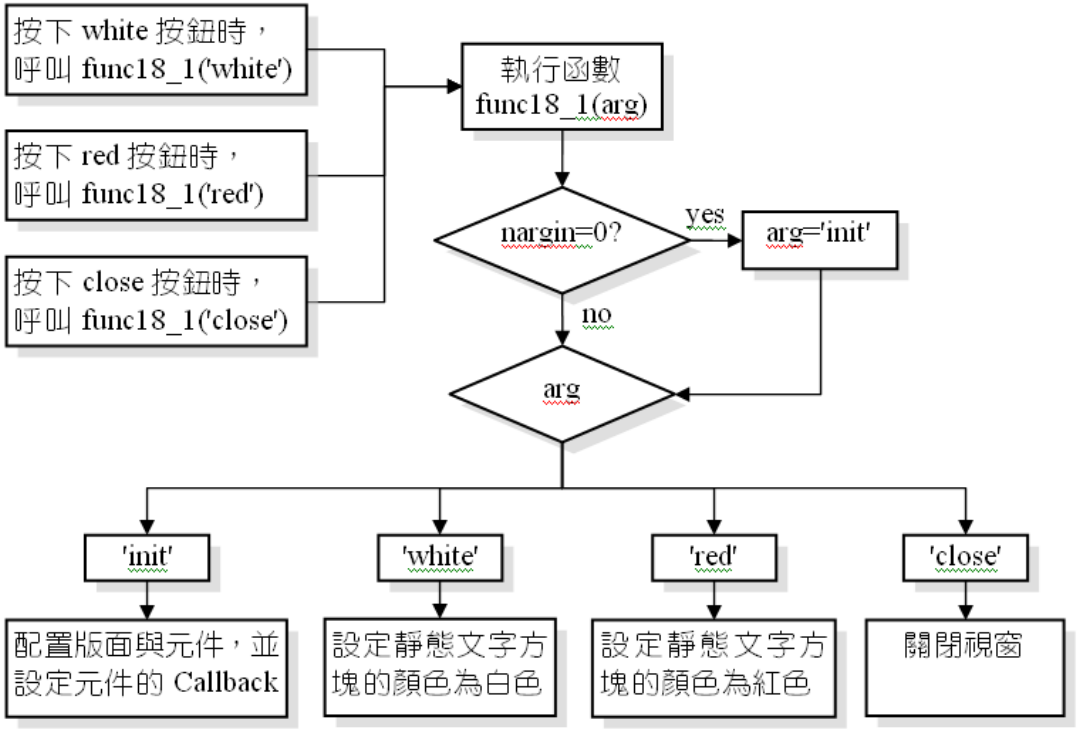
- 下面的範例修改自17.4.2節的M檔案script17_2.m，現在以switch-yard技術來改寫它：

按下 close 按鈕，則會跳出一個視窗，
用來確定是否真的要關閉

```
01  %func18_1.m, 簡單的事件處理範例，使用switch-yard技術
02  function func18_1(arg)
03
04  if nargin==0                % 如果輸入的引數個數為0，則設定arg='init'
05      arg='init';
06  end
07
08  switch(arg)
09   case 'init'                % 如果arg='init'，則執行下列的程式碼
10    figure('Position',[80 80 270 150],'Menubar','none');
11
12    h_close=uicontrol('String','close');
13    h_white=uicontrol('String','white','Position',[20 80 60 20]);
14    h_red=uicontrol('String','red','Position',[20 110 60 20]);
15    h_txt=uicontrol('Style','text','Position',[100 20 150 110]);
16
17    set(h_txt,'Tag','txt');
18    set(h_red,'Callback','func18_1 red');
19    set(h_white,'Callback','func18_1 white');
20    set(h_close,'Callback','func18_1 close');
```

```
21
22  case 'white'                % 如果arg='white'，則執行下列的程式碼
23    h=findobj(0,'Tag','txt');
24    set(h,'BackgroundColor',[1 1 1]);
25
26  case 'red'          % 如果arg='red'，則執行下列的程式碼
27    h=findobj(0,'Tag','txt');
28    set(h,'BackgroundColor',[1 0 0]);
29
30  case 'close'                % 如果arg='close'，則執行下列的程式碼
31    result=questdlg('確定要關閉?','Window closing','yes','no','no');
32    if strcmp(result,'yes')
33        close
34    end
35  end
```

- 下圖繪出了函數func18_1的執行流程：



18.1.3 使用global變數來存放handle

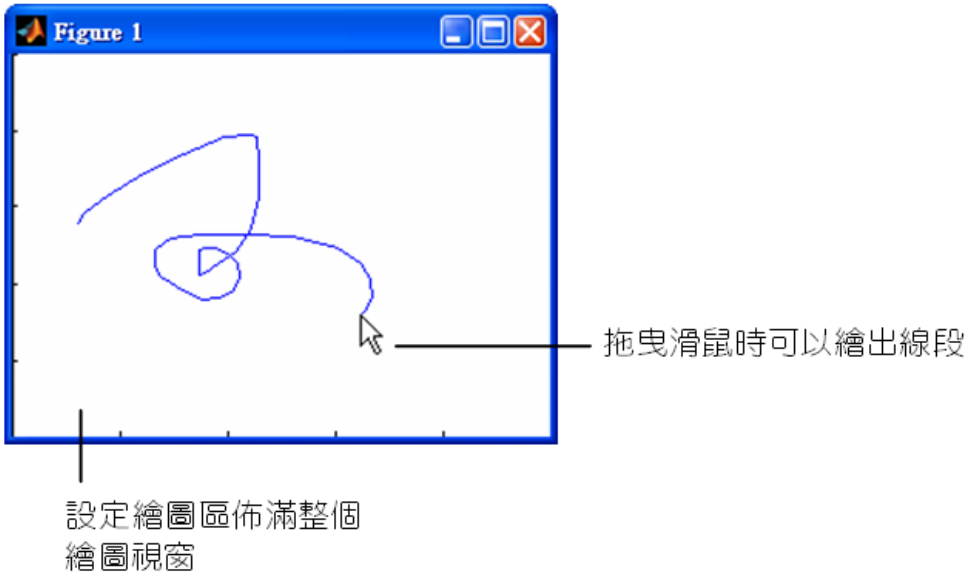
- 下面的範例同func18_1，但把變數h_txt 改以全域變數來設定：

```
%func18_2.m, 簡單的事件處理範例，使用全域變數來存放handle
function func18_2(arg)
global h_txt;                          % 宣告全域變數h_txt

if nargin==0
    arg='init';
end

switch(arg)
 case 'init'
  figure('Position',[80 80 270 150],'Menubar','none');
  h_close=uicontrol('String','close');
  h_white=uicontrol('String','white','Position',[20 80 60 20]);
  h_red=uicontrol('String','red','Position',[20 110 60 20]);
  h_txt=uicontrol('Style','text','Position',[100 20 150 110]);

```

```
17    set(h_txt,'Tag','txt');
18    set(h_red,'Callback','func18_2 red');
19    set(h_white,'Callback','func18_2 white');
20    set(h_close,'Callback','func18_2 close');
21
22  case 'white'                % h_txt為全域變數,所以可以直接取用
23    set(h_txt,'BackgroundColor',[1 1 1]);
24
25  case 'red'          % h_txt為全域變數,所以可以直接取用
26    set(h_txt,'BackgroundColor',[1 0 0]);
27
28  case 'close'
29   result=questdlg('確定要關閉?','Window closing','yes','no','no');
30   if strcmp(result,'yes')
31       close
32   end
33  end
```

18.1.4 利用switch-yard技術設計滑鼠事件

- 下面的範例修改自17.5.2節的例題，也就是拖曳滑鼠時可以繪出線段：



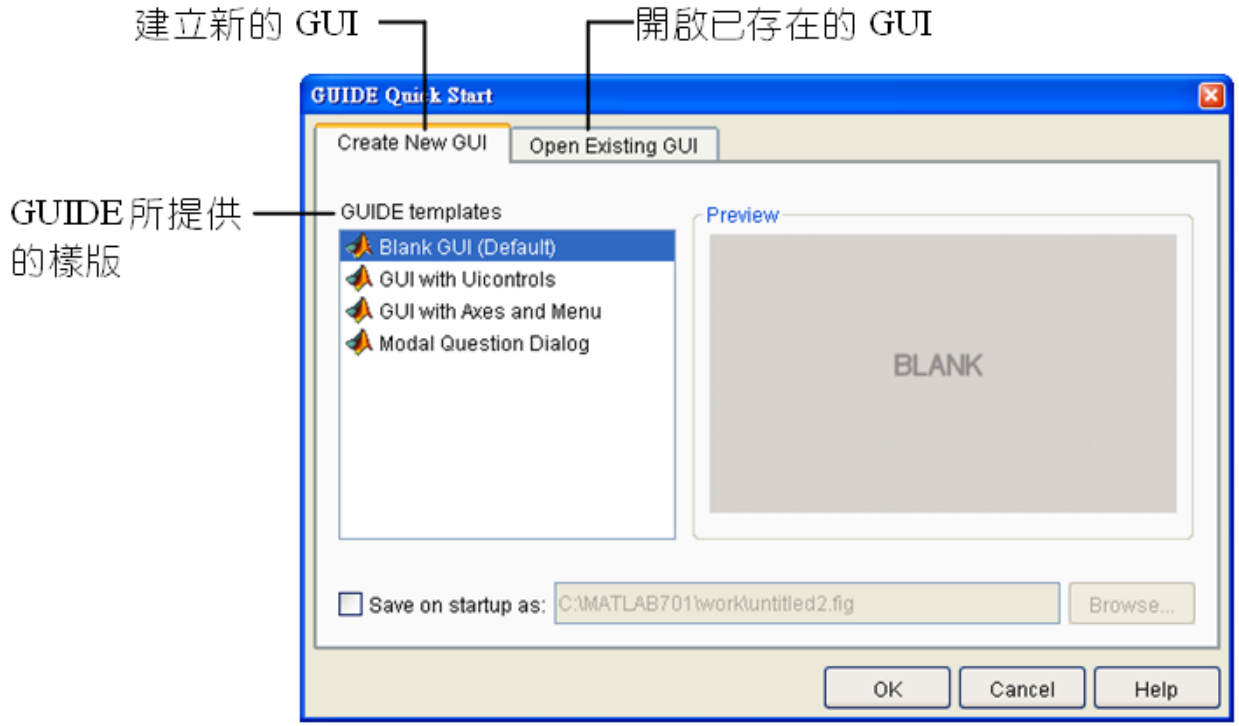
```
01  %func18_3.m, 滑鼠事件,使用switch-yard技術
02  function func18_3(arg)
03  global x0 y0 x1 y1;        % 設定全域變數
04
05  if nargin==0
06      arg='init';
07  end
08
09  switch(arg)
10  case 'init'
11      figure('Position',[80 80 280 200],'Menubar','none');
12      axes('Position',[0 0 1 1]);  % 設定繪圖區佈滿整個繪圖視窗
13      axis([0 1 0 1]);
14
15      set(gcf,'WindowButtonDownFcn','func18_3 down');
16      set(gcf,'WindowButtonUpFcn','func18_3 up');
17
18  case 'down'
19      current_pt=get(gca,'CurrentPoint');
20      x0=current_pt(1,1);
21      y0=current_pt(1,2);
22      set(gcf,'WindowButtonMotionFcn','func18_3 motion');
23
24  case 'motion'
```

```
    current_pt=get(gca,'CurrentPoint');
    x1=current_pt(1,1);
    y1=current_pt(1,2);
    line([x0,x1],[y0,y1]);
    x0=x1;
    y0=y1;

case 'up'
    set(gcf,'WindowButtonMotionFcn','');
end
```

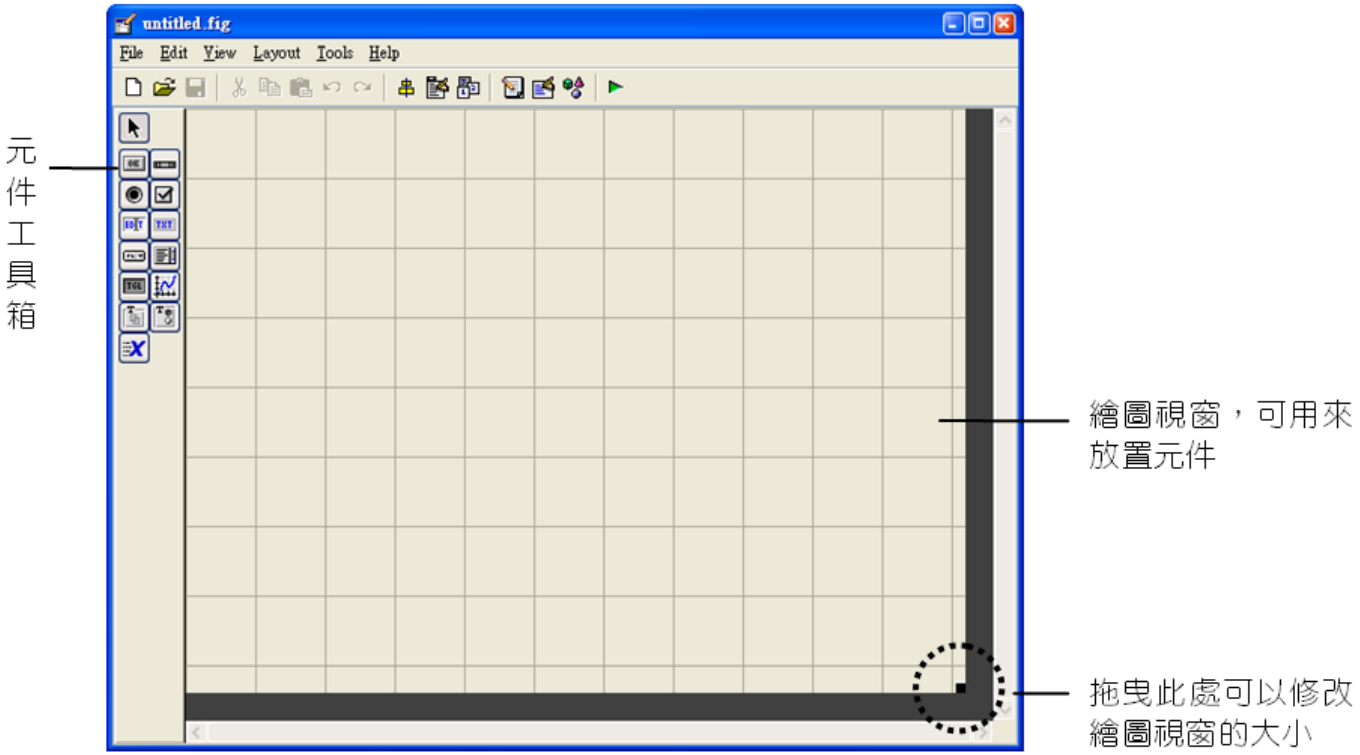
18.2 使用GUIDE設計GUI視窗介面

18.2.1 GUIDE 簡介

- 如要啟動GUIDE，請在工作視窗裡鍵入guide，此時的視窗如下圖：

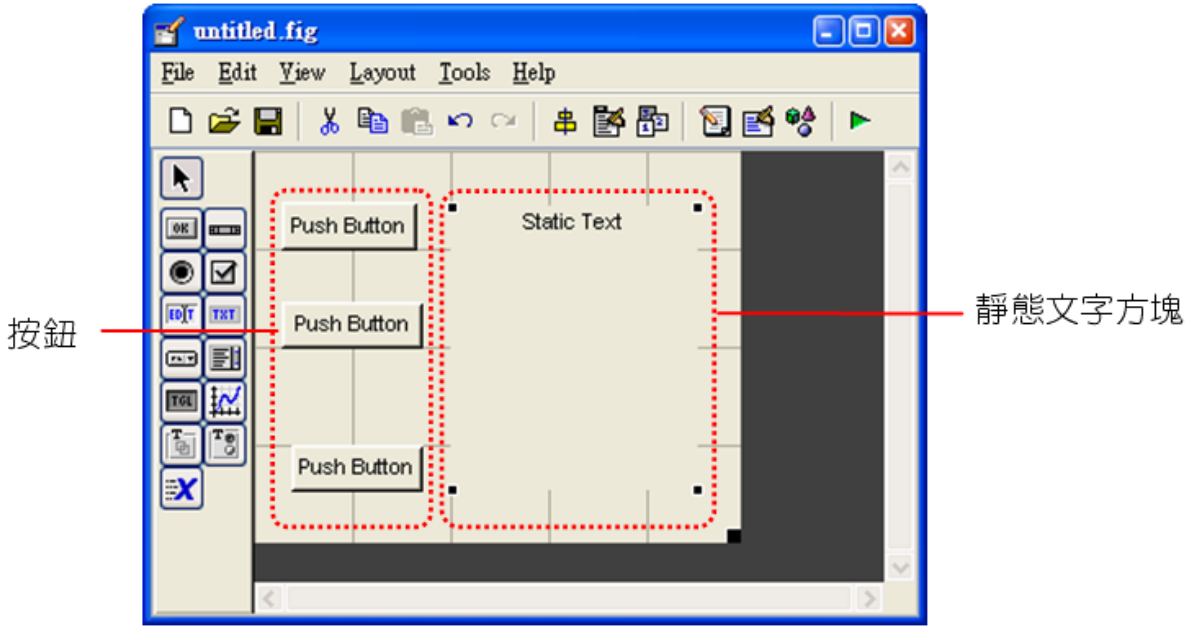


- 下面是空白GUI設計視窗：

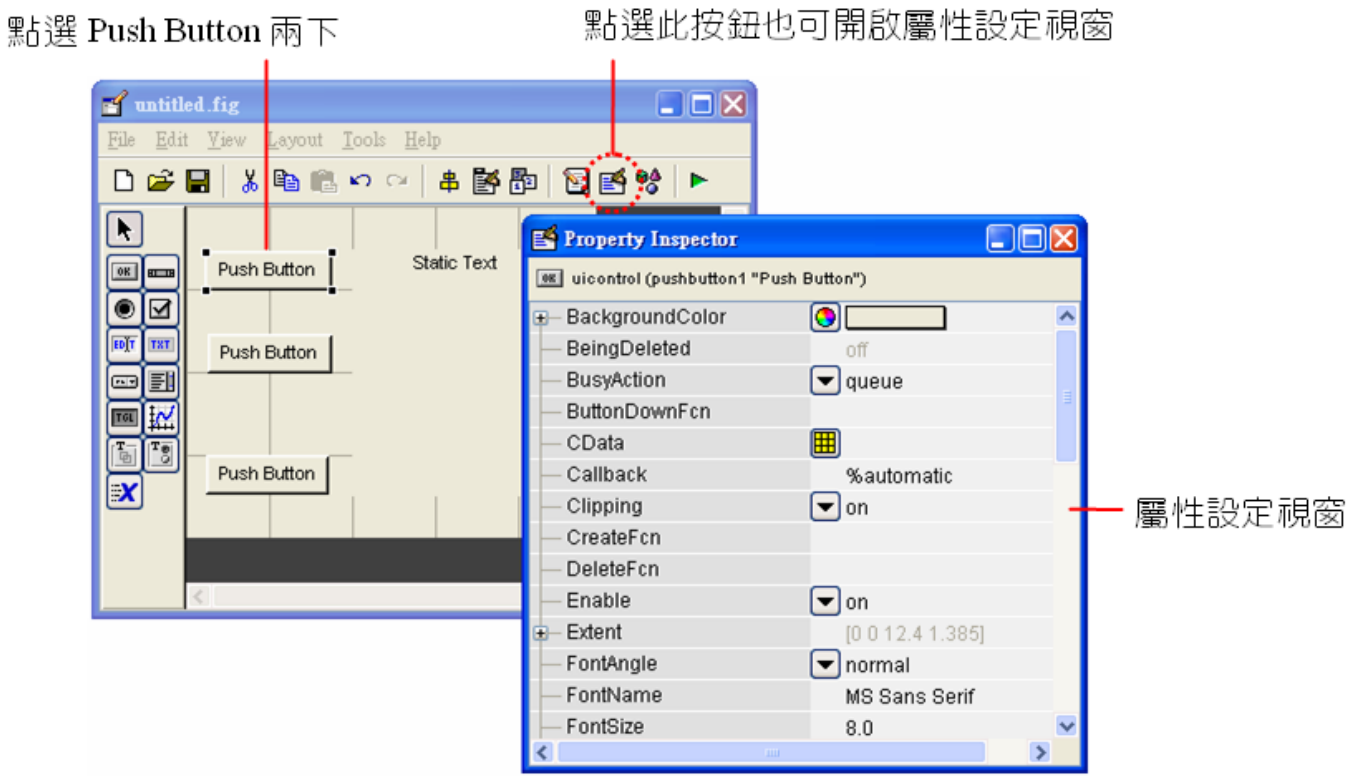


18.2.2 簡單的範例

- 請拉三個按鈕與一個文字方塊到繪圖視窗內，並佈置成如下的配置：

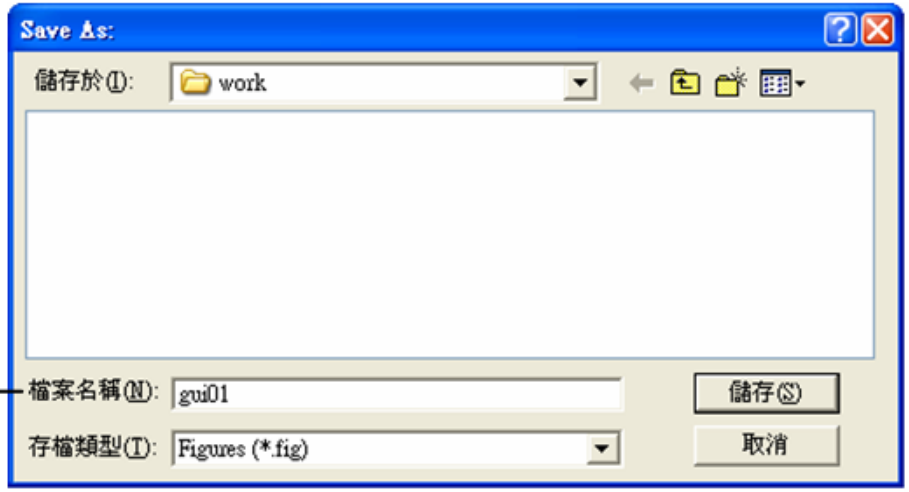


- 屬性設定視窗可用來設定與檢視元件的屬性：

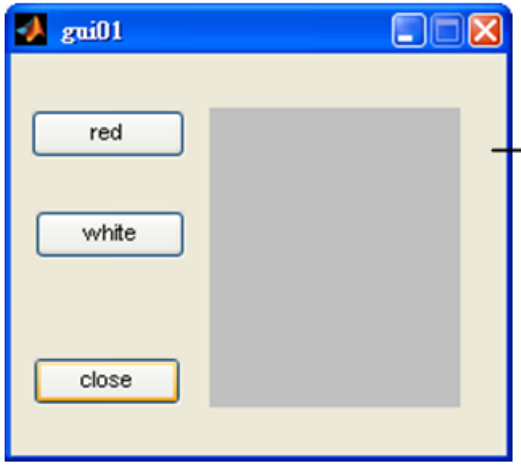


- 設計好應有的介面之後，便可將它存檔：

將檔案名稱設定為 gui01.fig



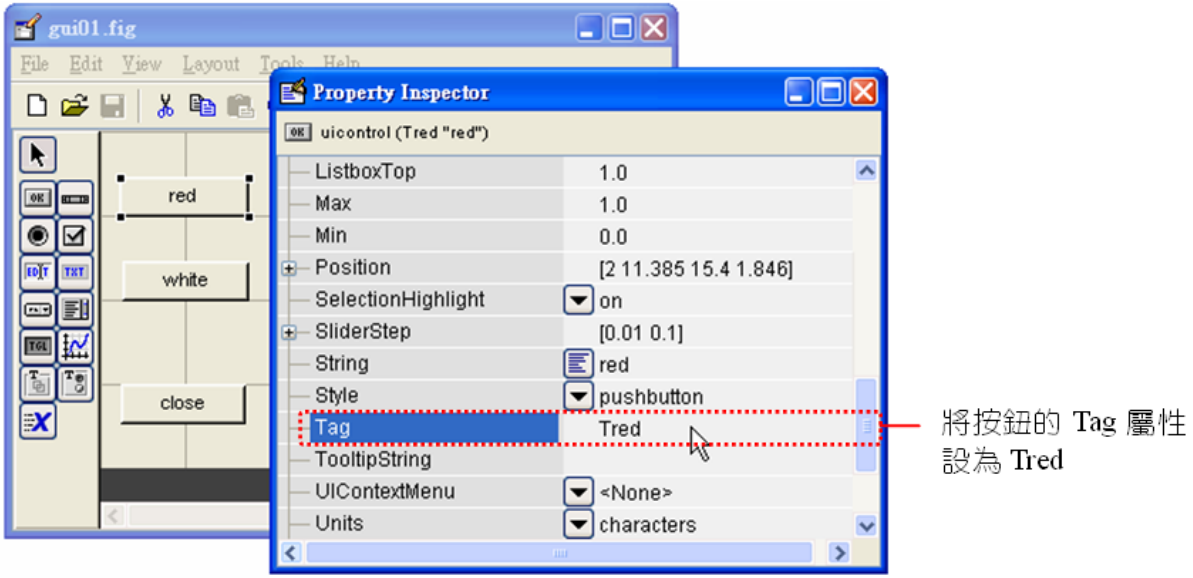
- 存檔完成後，Matlab即會執行它，如下面的畫面：



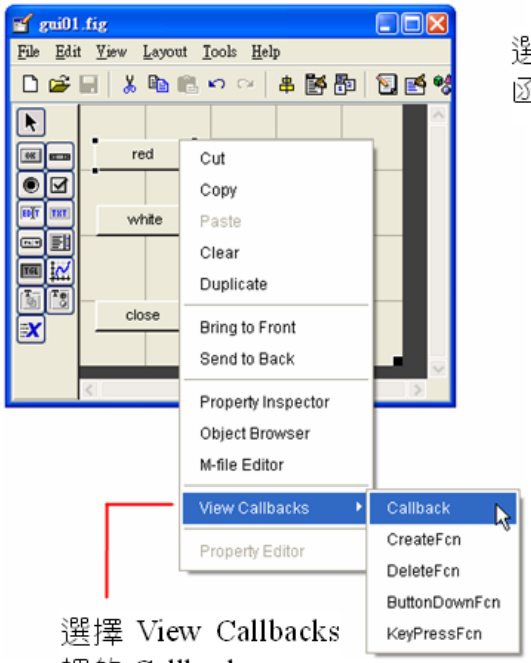
按下 ▶ 鈕之後的
執行結果

18.2.3 撰寫事件處理

- 要設定元件的Tag屬性，可於屬性設定視窗中設定：

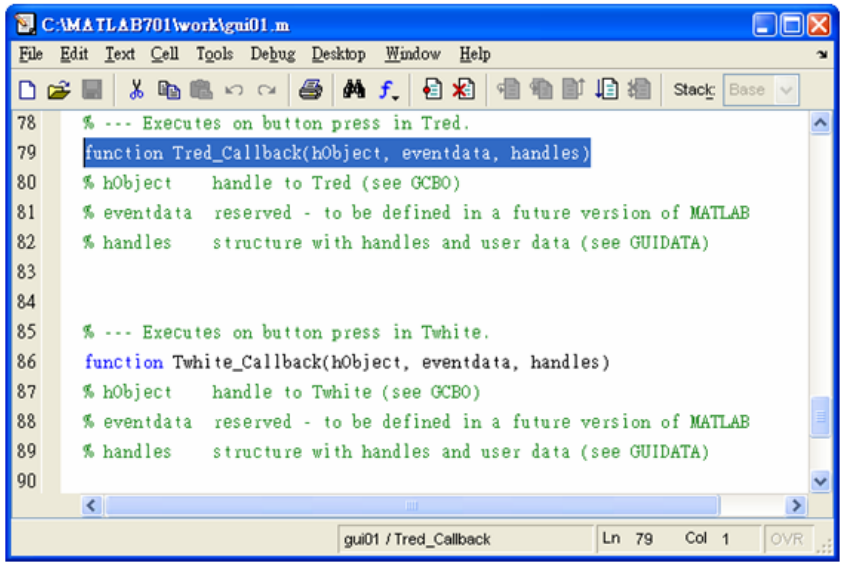


- 撰寫事件程式碼，請於出現的選單中選擇Callback：



選擇 View Callbacks 裡的 Callback

選擇 red 按鈕的 Callback 之後，會出現函數 Tred_Callback 的空殼讓您編輯



- red按鈕之反應函數每一個引數所代表的意義

保留給未來的版本使用
（目前的版本未使用到）

red 按鈕的 Tag 屬性

```
function Tred_Callback(hObject,eventdata,handles)
```

red 按鈕的反應函數

red 按鈕的 handle

由繪圖視窗內，所有元件之 handle 與使用者定義之資料所組成的結構

- red按鈕的事件處理：

```
% --- Executes on button press in Tred.
function Tred_Callback(hObject, eventdata, handles)
% hObject    handle to Tred (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
set(handles.Tcolor,'BackgroundColor',[1 0 0]);
```

- white按鈕的事件處理：

```
% --- Executes on button press in Twhite.
function Twhite_Callback(hObject, eventdata, handles)
% hObject    handle to Twhite (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
set(handles.Tcolor,'BackgroundColor',[1 1 1]);
```

- close按鈕的處理：

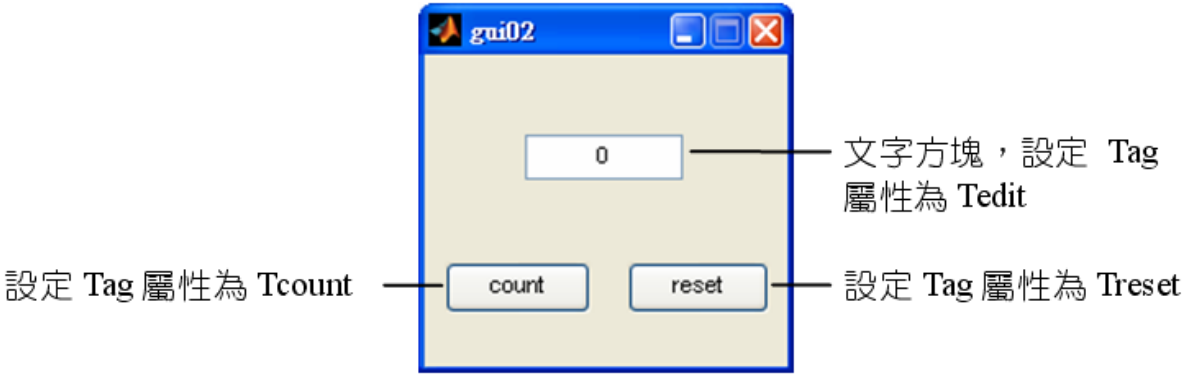
```
% --- Executes on button press in Tclose.
function Tclose_Callback(hObject, eventdata, handles)
% hObject    handle to Tclose (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
result=questdlg('確定要關閉?','Window closing','yes','no','no');
if strcmp(result,'yes')
    close
end
```

18.2.4 將資料儲存在handles結構中

- 利用下面的語法可把某個變數讓所有的元件共享：

```
handles.var_name=val;
guidata(hObject,handles);
```

- 下面是把某個變數讓所有的元件共享的範例：



在M檔案 gui02.m 裡，請把

```
handles.cnt=0;
```

這行程式碼寫在 gui02_OpeningFcn 函數裡：

```
% --- Executes just before gui02 is made visible.
function gui02_OpeningFcn(hObject, eventdata, handles, varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to gui02 (see VARARGIN)
handles.cnt=0;  % 設定cnt的值為0
% Choose default command line output for gui02
handles.output = hObject;
% Update handles structure
guidata(hObject, handles);
```

- 下面是按下 count 按鈕時，要處理的程式碼：

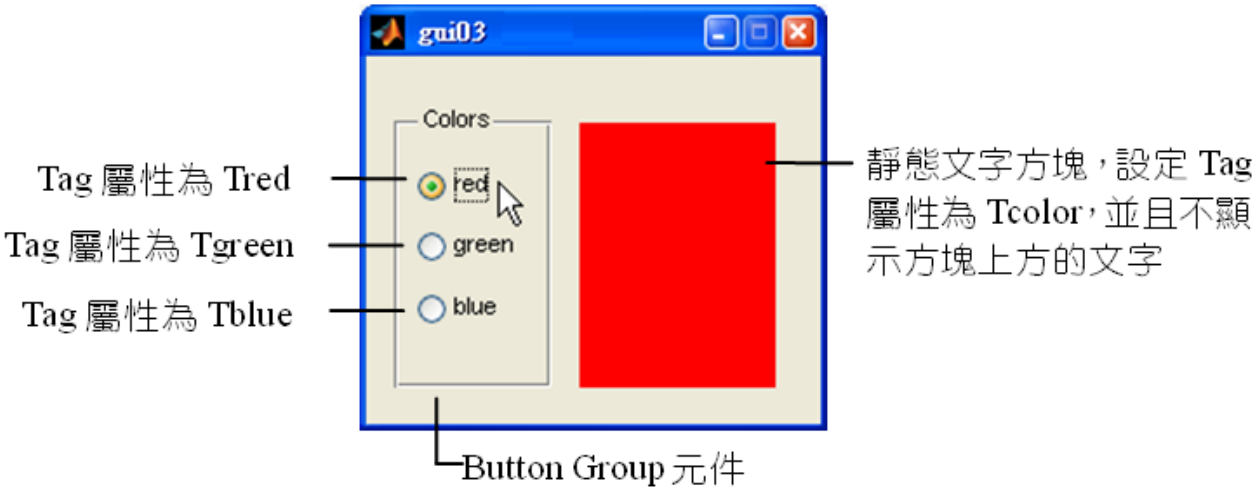
```
% --- Executes on button press in Tcount.
function Tcount_Callback(hObject, eventdata, handles)
% hObject    handle to Tcount (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
handles.cnt=handles.cnt+1;
set(handles.Tedit,'String',handles.cnt);
guidata(hObject, handles);
```

- 下面是按下 reset 鈕的事件處理：

```
% --- Executes on button press in Treset.
function Treset_Callback(hObject, eventdata, handles)
% hObject    handle to Treset (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
handles.cnt=0; % 設定cnt的值為0
set(handles.Tedit,'String',handles.cnt);
guidata(hObject, handles);
```

18.2.5 使用Button Group元件

- 只要放置在Button Group元件上方的選擇按鈕，就會自動設成互斥。
- 下面的是使用Button Group的簡單範例：

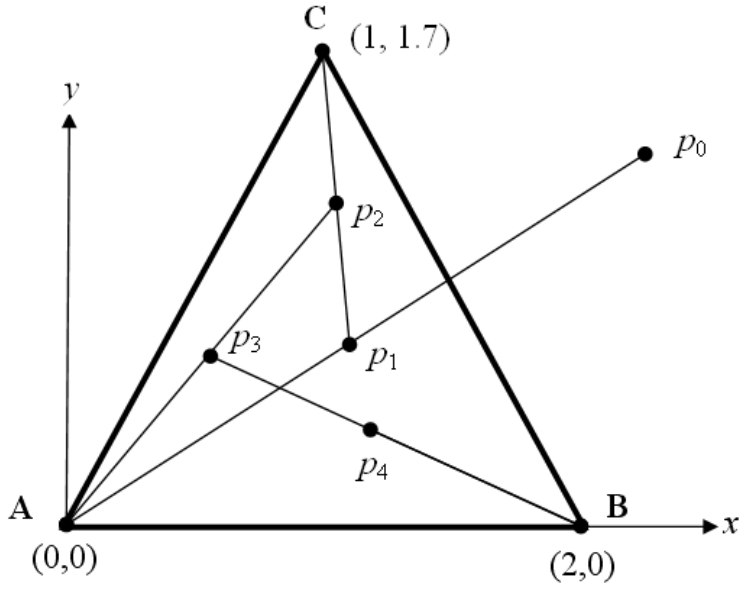


- 請在M檔案裡撰寫下面三行粗體字的程式碼：

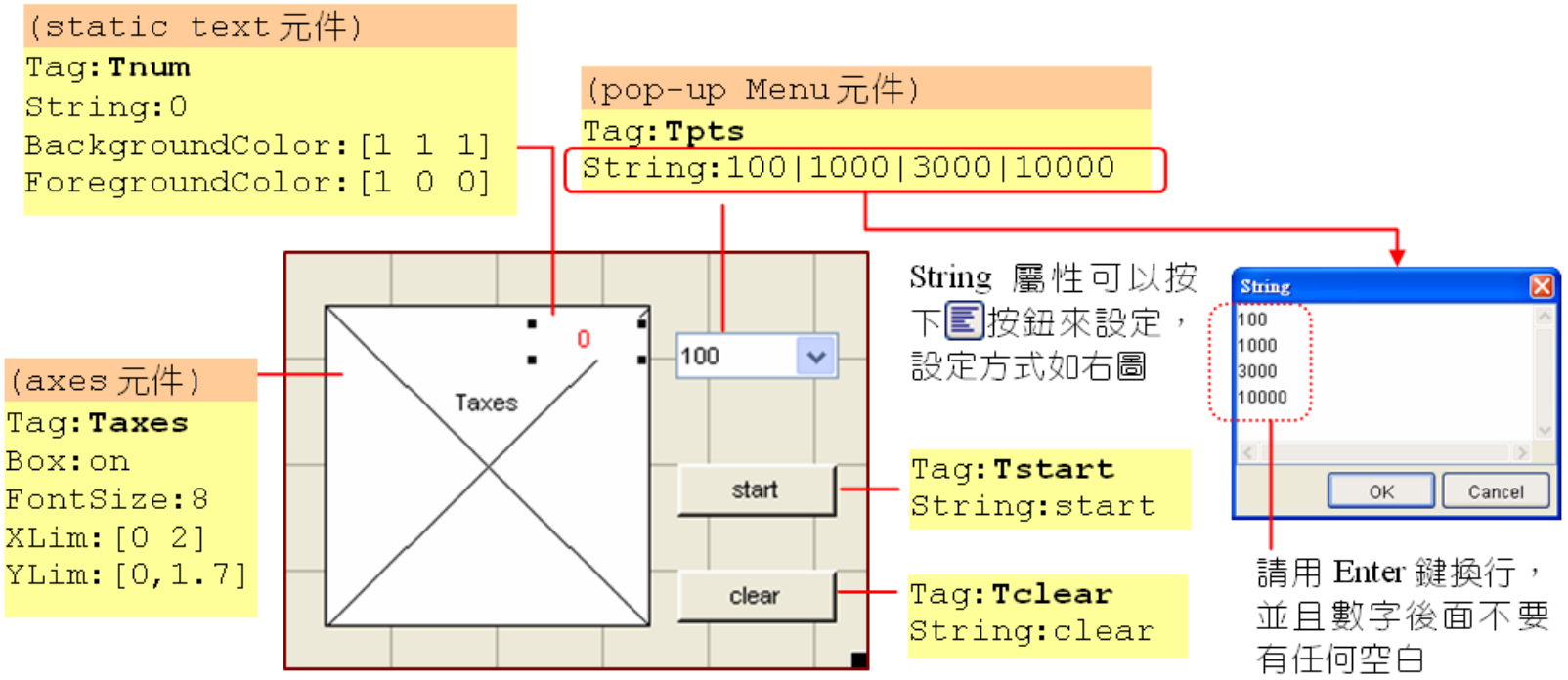
```
% --- Executes on button press in Tred.
function Tred_Callback(hObject, eventdata, handles)
% …
set(handles.Tcolor,'BackGroundColor',[1 0 0]);

 % --- Executes on button press in Tgreen.
function Tgreen_Callback(hObject, eventdata, handles)
% …
set(handles.Tcolor,‘BackGroundColor’,[0 1 0]);

% --- Executes on button press in Tblue.
function Tblue_Callback(hObject, eventdata, handles)
% …
set(handles.Tcolor,'BackGroundColor',[0 0 1]);
```

18.2.6 簡單的繪圖練習

- 下面的範例
  1. 介紹如何刪除已繪製的線段元件
  2. 探討如何在迴圈裡，使得GUI元件得以持續更新



- 本範例元件的屬性（已標示在元件的旁邊）：



- 請在gui04_OpeningFcn函數裡加入程式碼：

```
function gui04_OpeningFcn(hObject, eventdata, handles, varargin)
% …
handles.flag=0;                    % 設定flag變數為0
% Choose default command line output for chaotic
handles.output = hObject;
```

- 撰寫按下start按鈕時所要執行的程式碼：

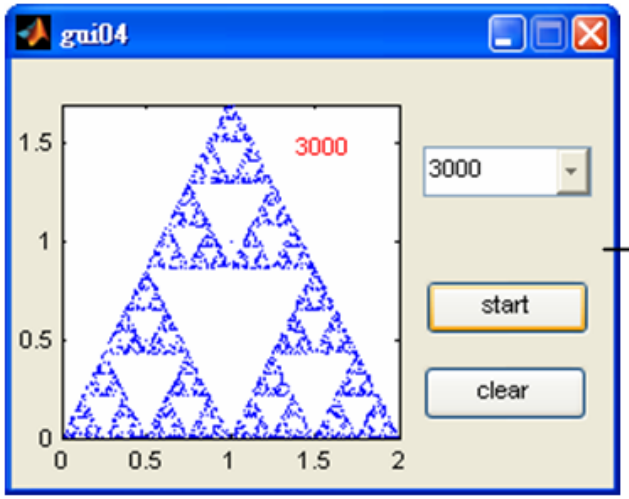
```
% --- Executes on button press in Tstart.
function Tstart_Callback(hObject, eventdata, handles)
% …
if handles.flag==1                 %flag==1代表繪圖區已繪有圖形
delete(handles.h);                 %刪除所有的資料點
```

```
% --- Executes on button press in Tstart.
function Tstart_Callback(hObject, eventdata, handles)
if handles.flag==1
delete(handles.h);
guidata(hObject,handles);
end

x0=1;
y0=1;
v=[0 0;1 sqrt(3);2 0];
hold on;

str=get(handles.Tpts,'String');  % 取得下拉選單裡的所有字串
val=get(handles.Tpts,'Value');   % 取得下拉選單被選取選項的索引值
n_pts=str2num(str{val});
handles.h=[];  % 將陣列h的內容清空

for i=1:n_pts
   set(handles.Tnum,'String',i);      % 設定文字方塊顯示的數值為i
   handles.h(i)=plot(x0,y0);  % 繪出點(x0,y0)，並設定其handle給h
   pt=v(ceil(rand()*3),:);    % 隨機取出一個頂點
   x0=(x0+pt(1))/2;
   y0=(y0+pt(2))/2;
end

handles.flag=1;               % 設定flag為1，代表繪圖區內已繪點
guidata(hObject,handles);
```

- 接下來是clear按鈕的事件處理：

```
% --- Executes on button press in Tclear.
function Tclear_Callback(hObject, eventdata, handles)
if handles.flag==1                          % 如果繪圖區裡已有資料
   delete(handles.h);                       % 清空繪圖區裡的資料點
   set(handles.Tnum,'String',0);            % 設定文字方塊顯示的數值為0
   handles.flag=0;                          % 將flag的值設為0
   guidata(hObject,handles);
end
```

- 下圖是本範例執行的結果：



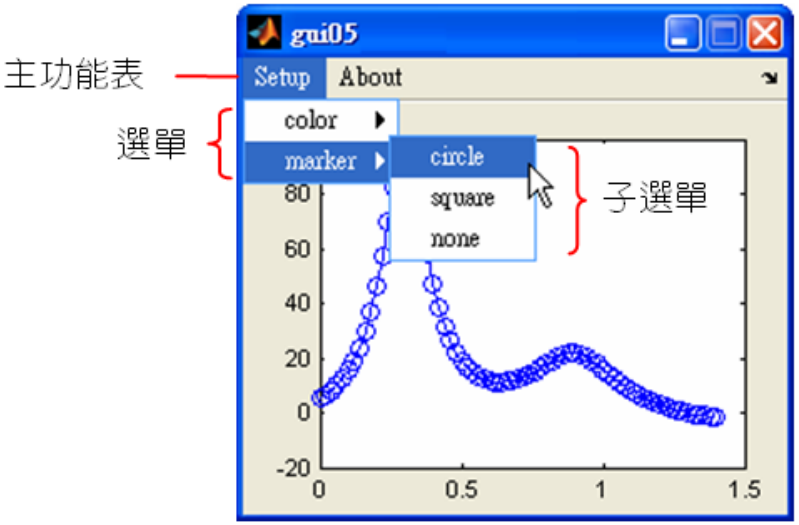
當程式正在繪圖時，請先不要觸動視窗裡其它的元件，以避免更動到某些變數而導致錯誤產生

- 如果想強迫圖形元件在迴圈內立即更新，可以利用drawnow指令：

```
for i=1:n_pts
set(handles.Tnum,'String',i);
handles.h(i)=plot(x0,y0);
if(mod(i,50)==0)
   drawnow
end
pt=v(ceil(rand()*3),:);
x0=(x0+pt(1))/2;
y0=(y0+pt(2))/2;
end
```

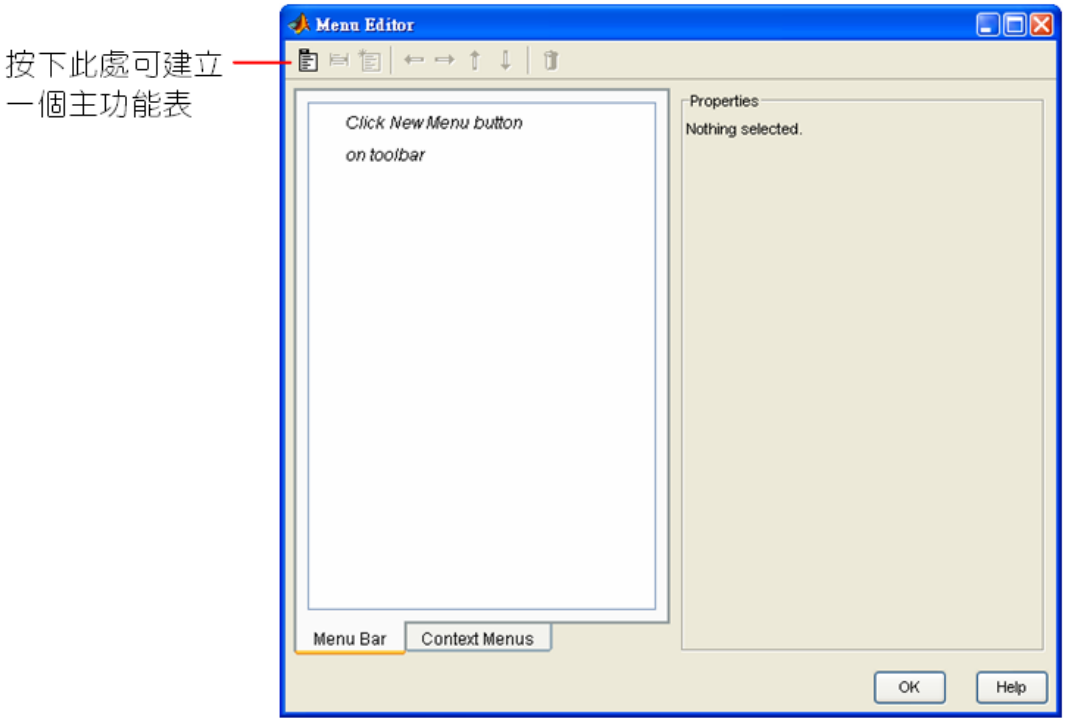
18.3 功能表的設計

- 下面是功能表使用方式的範例：

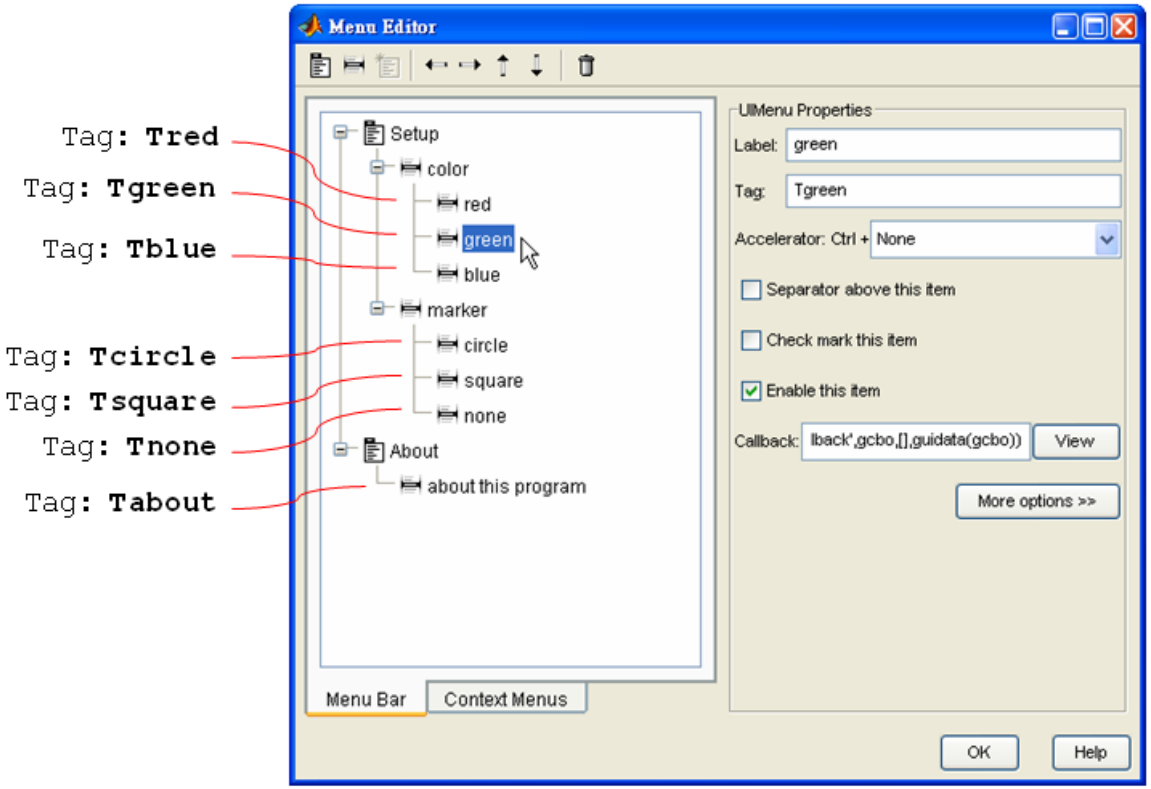


- 在工具列裡按下Menu Editor按鈕 

  此時Menu Editor視窗會出現，如下圖所示：



- 下圖是配置好選單與Tag 屬性之視窗：



- 把繪製圖形的函數撰寫在gui05_OpeningFcn裡：

```
% --- Executes just before gui05 is made visible.
function gui05_OpeningFcn(hObject, eventdata, handles, varargin)
[x,y]=humps(0:0.02:1.4);  % 計算humps函數的值
handles.h=plot(x,y);       % 繪出humps函數，並把線段的handle設給h
```

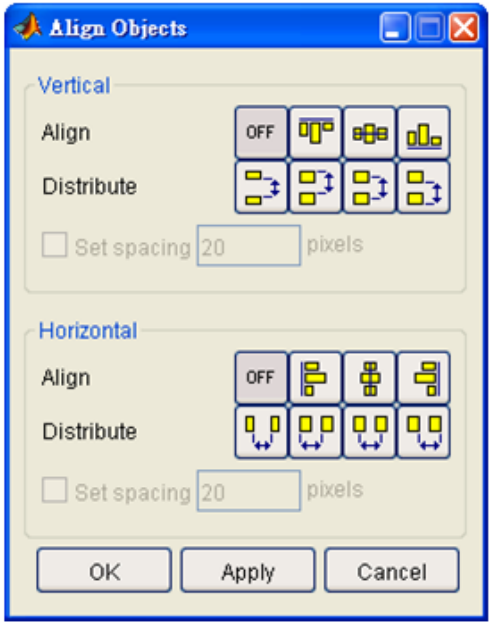
- 接下來撰寫在功能表裡每一個選單的事件處理：

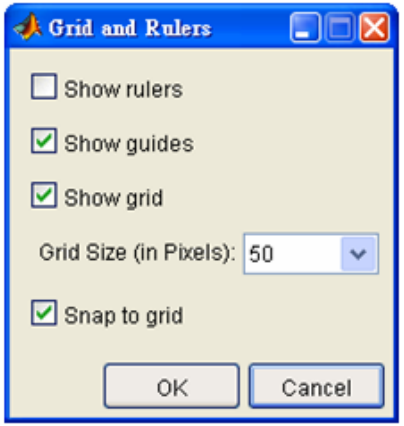
```
function Tred_Callback(hObject, eventdata, handles)
set(handles.h,'Color',[1 0 0]);        % 將線段顏色設定為紅色

function Tgreen_Callback(hObject, eventdata, handles)
set(handles.h,‘Color’,[0 1 0]);        % 將線段顏色設定為綠色

function Tblue_Callback(hObject, eventdata, handles)
set(handles.h,‘Color’,[0 0 1]);        % 將線段顏色設定為藍色

function Tcircle_Callback(hObject, eventdata, handles)
set(handles.h,‘Marker’,‘o’);           % 將標識符號設定為小圓圈

function Tsquare_Callback(hObject, eventdata, handles)
set(handles.h,‘Marker’,‘s’);           % 將標識符號設定為正方形

function Tnone_Callback(hObject, eventdata, handles)
set(handles.h,‘Marker’,‘none’);        % 不使用標識符號

function Tabout_Callback(hObject, eventdata, handles)
msgbox('A GUI menu test program');            % 跳出訊息視窗
```

18.4 GUIDE其它常用的功能

- Align Object對話方塊與Grid and Rulers對話方塊：

Align Object 對話方塊，可用來對齊元件的佈置





Grid and Rulers 對話方塊可設定是否顯示尺規與其它項目

-The End-