

第七章 图形用户界面 (GUI) 的设计

Graphical user interfaces

7.1 GUI概述

1. 图形用户界面（GUI）

- **GUL**指由窗口、菜单、图标、光标、按键、对话框和文本等各种图形对象组成的用户界面。它让用户定制用户与Matlab的交互方式，而命令窗口不是唯一与Matlab的交互方式。
- 用户通过鼠标或键盘选择、激活这些图形对象，使计算机产生某种动作或变化。

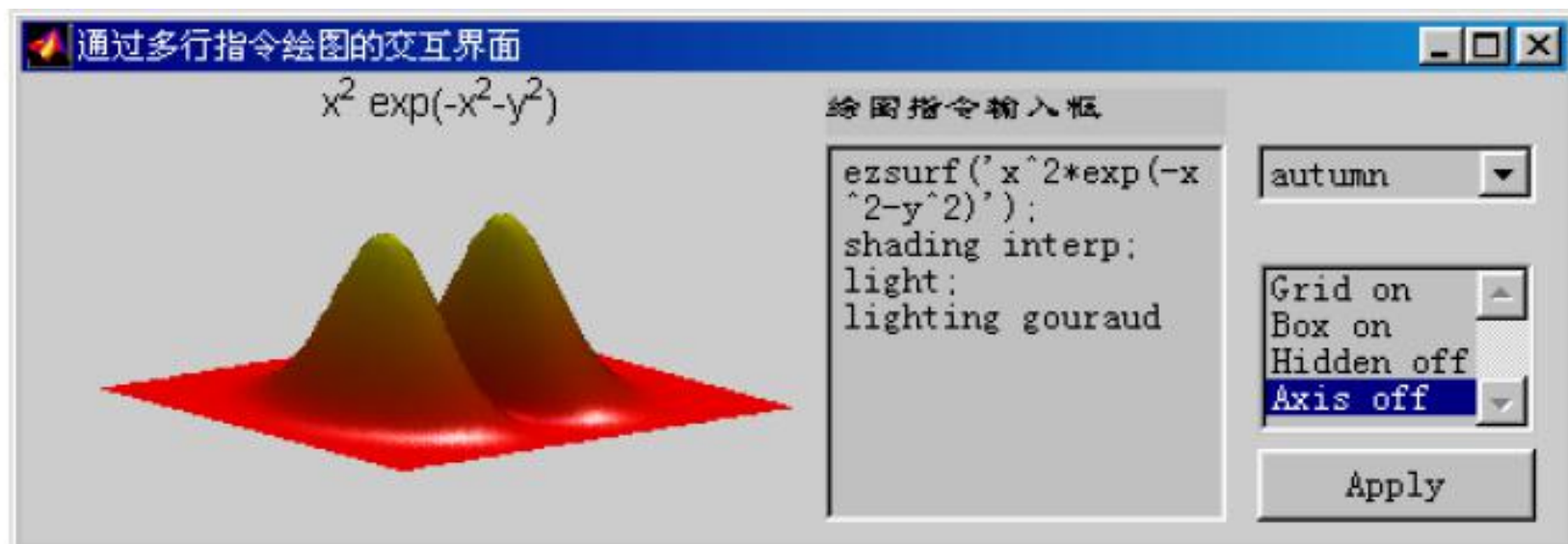
2. 图形用户界面的设计方法

一般有两种方法:

①编写程序:直接通过编辑M脚本文件产生GUI;

②使用可视化的界面环境GUIDE:

通过Matlab图形用户界面开发环境GUIDE(Graphical User Interface Development Environment)来形成相应文件.



7.2 编程设计GUI

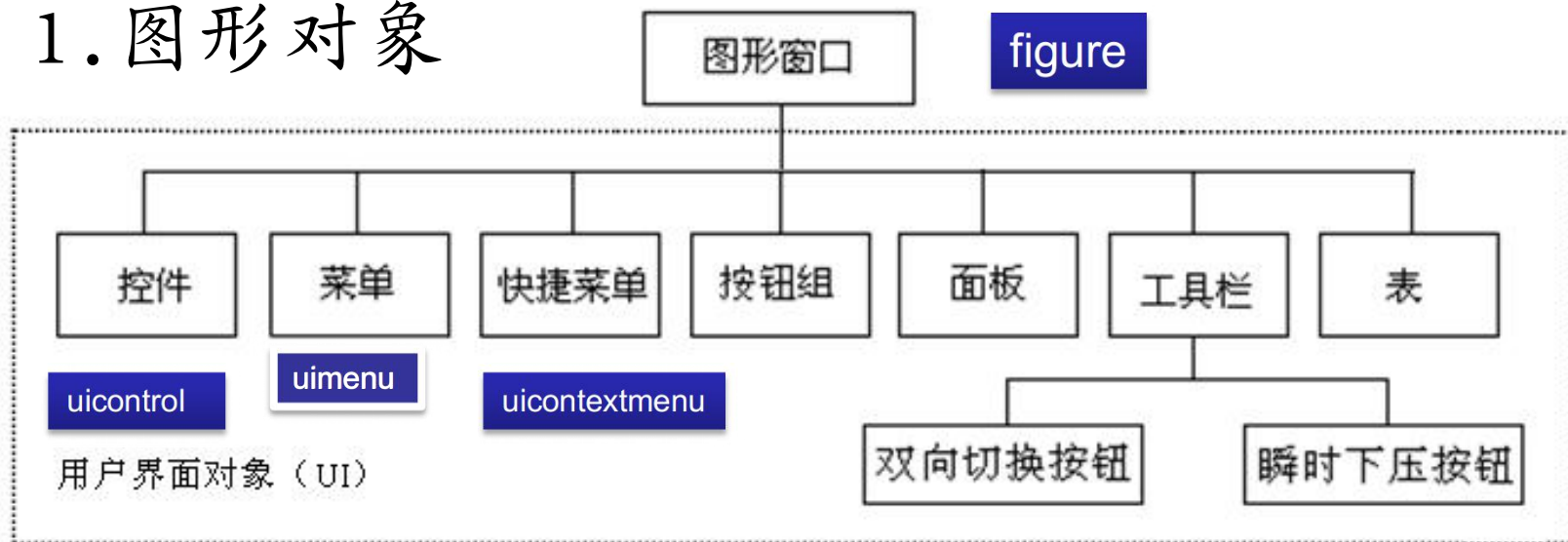
7.2.1 图形用户界面对象和句柄

7.2.2 菜单设计

7.2.3 对话框设计

7.2.1 图形用户界面对象和句柄

1. 图形对象



2. 图形句柄

图形句柄：创建图形对象时，都为该图形对象分配唯一的一个值，称其为图形句柄 (**handle**)。句柄是图形对象唯一的标识符，不能重复和混淆。

3.图形句柄的常见函数

- 1.figure：创建一个新的图形窗口对象
- 2.set：设置图形对象的各项属性
- 3.get：获取图形对象的各项属性
4. uimenu：创建图形窗口的一级子菜单和子菜单
- 5.uicontextmenu:创建快捷菜单
6. uicontrol：生成用户控制图形对象
- 7.uibuttongroup:创建按钮组，管理单选和开关按钮
- 8.uipushtool:创建工具栏按钮
- 9.Uitoolbar；创建工具栏
10. axes：创建坐标轴图形对象
- 11 · line：画线。
- 12 · patch：填充多边形。
- 13 · surface：绘制三维曲面。
- 14.image：显示图片对象

4.程序编写GUI的步骤

第一步：建立figure对象以作为整个GUI的基础窗口，并设置该窗口的相关属性。若后续需要建立uimenu，则必须更改窗口的Menubar属性为menubar或figure。

第二步，决定建立需要的axes或uicontrol、uimenu、uicontextmenu对象，并设置相关属性及Callback。

注意：在应用uicontrol，uimenu等语句中，如果当前只有一个绘图窗口，则不必指定窗口；若有多个绘图窗口，则必须在uicontrol对象内指定显示在哪个窗口中。

5. 创建图形主界面（窗口）

(1). 建立图形窗口

窗口句柄=figure(属性名1，属性值1，属性名2，属性值2，...)

图形窗口的常用属性：

Name：图形窗口的标题，取值可以是任何字符串，它的缺省值为空。一般情况下，其标题形式为：Figure No.1:字符串；

NumberTitle：决定着在图形窗口的标题中是否以“Figure No.n:”为标题前缀，这里n是图形窗口的序号，即句柄值。属性的取值是on(缺省值)或off_{DEL}。

MenuBar：用来控制图形窗口是否应该具有菜单条。如果它的属性值为none，则表示该图形窗口没有菜单条。MenuBar属性的取值可以是figure(缺省值)，为标准菜单。

图形窗口的标准菜单：文件File、编辑Edit、视图View、插入Insert、工具Tools、窗口Windows和帮助Help七个菜单。每个都有下拉菜单Pull-down menu。也可以采用

uimenu函数在原默认的图形窗口菜单后面添加新的菜单项。

【例】 建立一个图形窗口。该图形窗口起始于屏幕左下角、宽度和高度分别为300像素点和150像素点，背景颜色为绿色，且当用户从键盘按下任意一个键时，将显示“Hello,Keyboard Key Pressed.”字样。

```
hf=figure('Color',[0,1,0],'Position',[1,1,300,150],...  
          'Name','图形窗口示例'  
, 'NumberTitle','off','MenuBar','none',...  
          'KeyPressFcn','disp(''Hello,Keyboard Key'  
Pressed.'')');
```



6. 图形对象属性的设置和获取

(1) 设置图形句柄属性函数set

- **set**(图形句柄，属性名1，属性值1，属性名2，属性值2，...)

(2) 获取图形句柄信息函数get

- **V=get**(句柄名，属性名)

- 其中V是返回的属性值。

函数名	功能描述
gca	获得当前坐标轴对象的句柄
gcbf	获得当前正在执行调用的图形对象的句柄
gcbo	获得当前正在执行调用的对象的句柄
gcf	获得当前图形对象的句柄
gco	获得当前对象的句柄

7.2.2 菜单设计

1. 建立用户菜单uimenu

该函数可以用于建立一级菜单项和子菜单项。

建立一级菜单项的函数调用格式为：

一级菜单项句柄=uimenu(图形窗口句柄，属性名1，属性值1，属性名2，属性值2，...)

建立子菜单项的函数调用格式为：

子菜单项句柄=uimenu(一级菜单项句柄，属性名1，属性值1，属性名2，属性值2，...)

2. 隐藏和显示标准菜单

创建图形窗口

```
h=figure
```

隐去标准菜单使用命令：

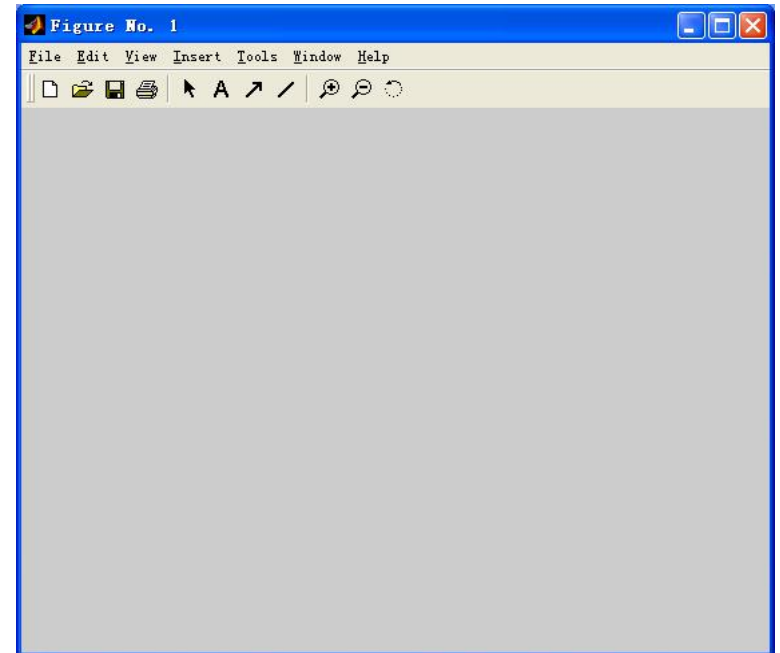
```
set(h,'MenuBar','none');  
set(gcf,'menubar','none');
```

恢复标准菜单使用命令：

```
set(gcf,'menubar','figure')
```



`set(h,'MenuBar','none');`



`set(gcf,'menubar','figure')`

3. 自制的用户菜单

本例演示：自制一个带下拉菜单表的用户菜单。该菜单能使图形窗背景颜色设置为兰色或红色，并且产生带分格的封闭坐标轴。

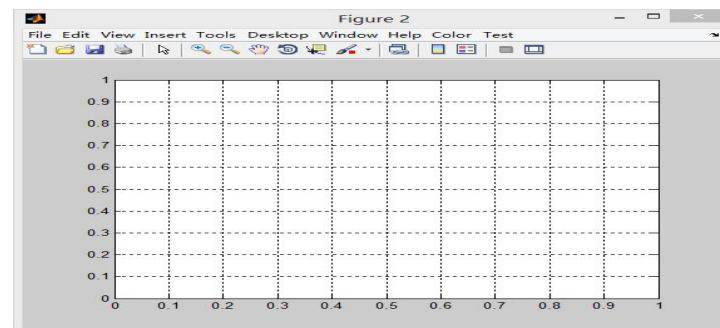
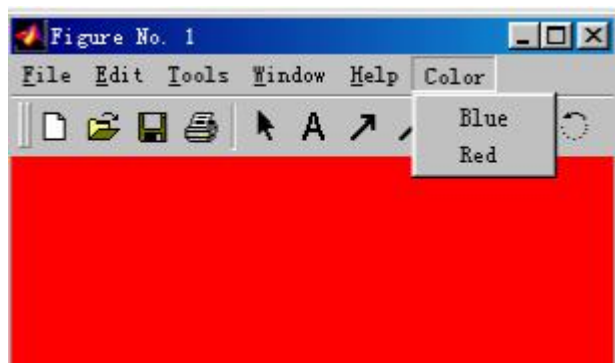
figure %创建一个图形窗口

h_menu=uimenu(gcf,'label','Color'); %制作用户顶层菜单项Color <2>

h_submenu1=uimenu(h_menu,'label','Blue',... %制作下拉菜单项Blue <3>
'callback','set(gcf,"Color","blue")'); %<4>

h_submenu2=uimenu(h_menu,'label','Red',... %制作下拉菜单Red <5>
'callback','set(gcf,"Color","red")');

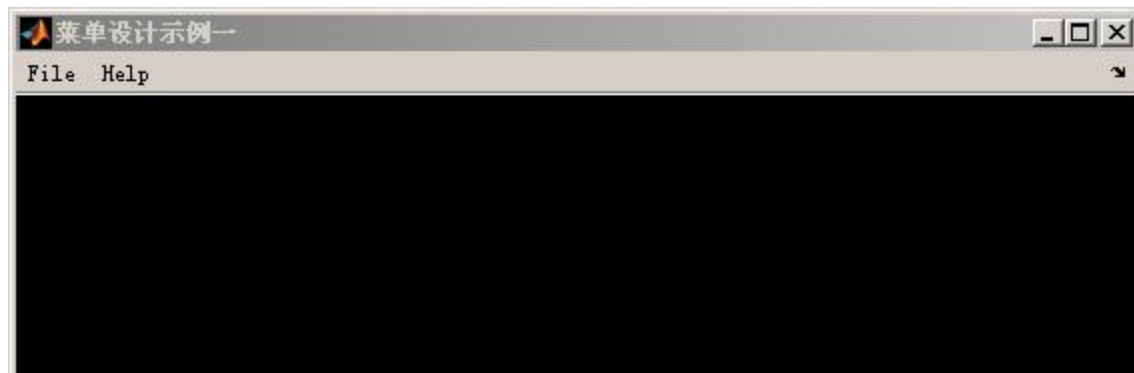
>> uimenu(gcf,'Label','Test','Callback','grid on,set(gca,"box","on"),')%可产生带分格的封闭坐标轴的test



【例】 建立一个菜单系统。

菜单条中含有File和Help两个菜单项。如果选择File中的New选项，则将显示New Item字样，如果选择File中的Open选项，则将显示出Open Item字样。File中的Save菜单项初始时处于禁选状态，在选择Help选项之后将此菜单项恢复成可选状态，如果选择File中的Save选项，则将出现一个新的菜单(三级菜单)，其中共有两个子菜单项Text file和Graphics file，如果选择第1项，则将变量k1和k2分别赋为0和1，然后调用file01.m文件来进行相应的处理(该文件需要另行编写)，如果选择第2项，则将变量k1和k2分别赋为1和0，然后调用file10.m文件来进行相应的处理(该文件也需要另行编写)。如果选择File中的Save As选项，则将显示Save As Item字样。如果选择File中的Exit选项，则将关闭当前窗口。如果选择Help中About ...选项，则将显示Help Item字样，并将Save_{DEL}菜单设置成可选状态。

```
hf=figure('Color',[0,1,1],'Name','菜单设计示例一',...  
'NumberTitle','off','MenuBar','none');  
    hfile=uimenu(hf,'label','&File');  
    hhhelp=uimenu(hf,'label','&Help');  
        uimenu(hfile,'label','&New','call','disp(''New Item'')');  
        uimenu(hfile,'label','&Open','call','disp(''Open Item'')');  
    hsave=uimenu(hfile,'label','&Save','Enable','off');  
    uimenu(hsave,'label','Text file','call','k1=0;k2=1;file01;');  
    uimenu(hsave,'label','Graphics file','call','k1=1;k2=0;file10;');  
    uimenu(hfile,'label','Save &As','call','disp(''Save As Item'')');  
    uimenu(hfile,'label','&Exit','separator','on','call','close(hf)');  
    uimenu(hhhelp,'label','About ...','call',...  
        ['disp(''Help Item'');','set(hsave,'Enable','on')']);
```



4. 设置简捷键或快捷键

设置简捷键或快捷键

简捷键的制作方法：

如果想为某菜单项制作简捷键（Shortcut key），只要使'Label'的属性值字符串包含&X 便可。在此的X代表用户喜欢的任何字母。

简捷键的提示方法：

被设置简捷键的那菜单名中将出现字符X。

简捷键的操作方式：

只有在相应菜单项可见时，简捷键操作（ALT+X）才起作用。

快捷键的制作方法：

某菜单项快捷键（Accelerator key）的制作，必须依赖用户菜单uimenu的‘Accelerator’属性，属性值可以取任何字母。

快捷键的提示方式：

被设置快捷键的那菜单名后将出现（Ctrl+Y）。

快捷键的操作方式：

当相应菜单项不可见时，快捷键操作才起作用。

简捷键主要用于顶层菜单项；快捷键主要用于（自身不再带子菜单的）子菜单项。

【*例】本例目标：Color菜单项及其下拉的Blue菜单各带一个简捷键，而另一项下拉菜单Red带一个快捷键。

```
hi=figure
```

```
h_menu=uimenu(hi,'Label','&Color'); %带简捷键C的用户菜单Color  
<2>
```

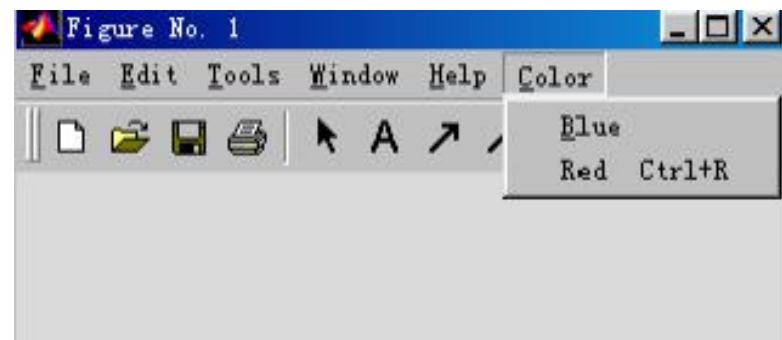
```
h_submenu1=uimenu(h_menu,'Label','&Blue',... %带简捷键B的的下拉  
菜单Blue <3>
```

```
'Callback','set(gcf,"color","blue")');
```

```
h_submenu2=uimenu(h_menu,'label','Red',... %制作另一个下拉菜单  
Red
```

```
'Callback','set(gcf,"color","red")',...
```

```
'Accelerator','r'); %为Red菜单设置快捷键R <7>
```



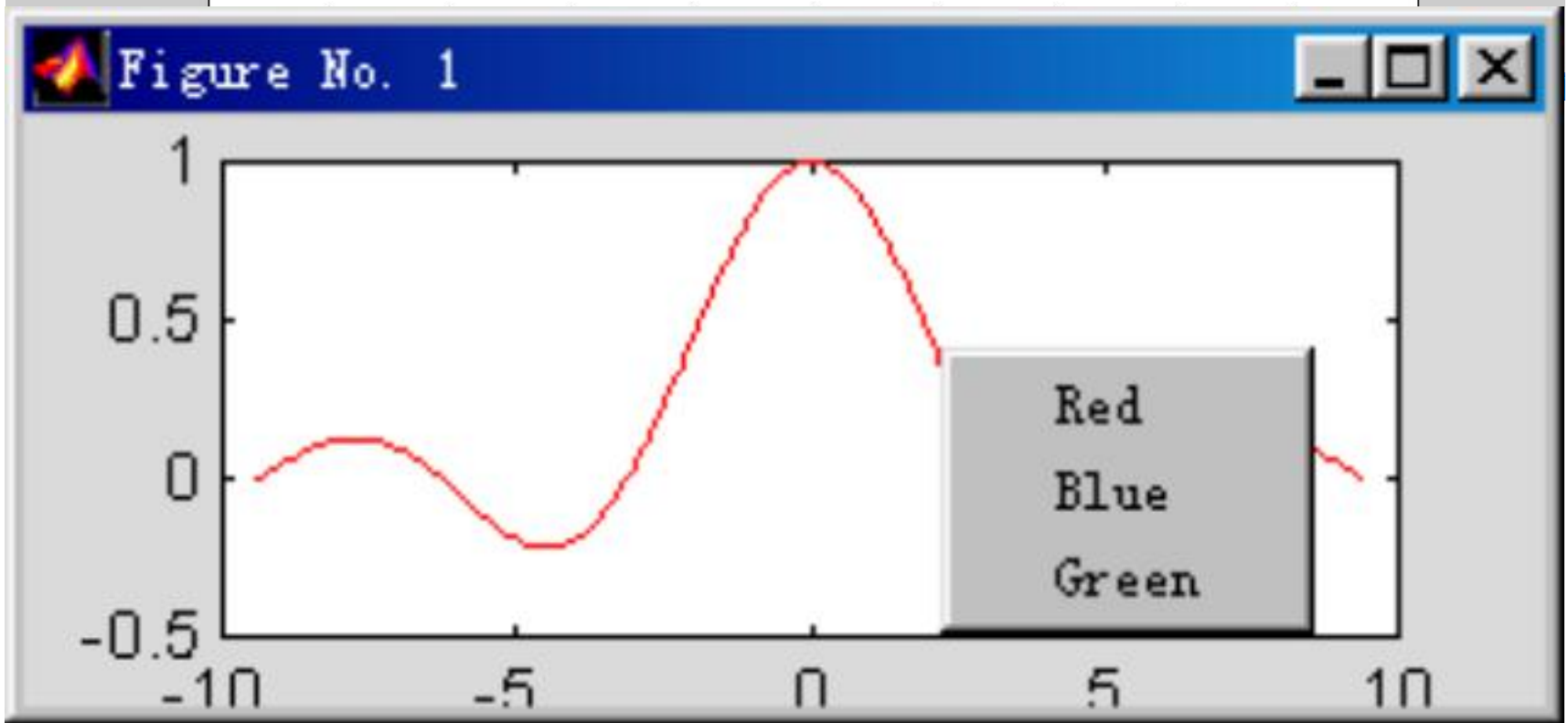
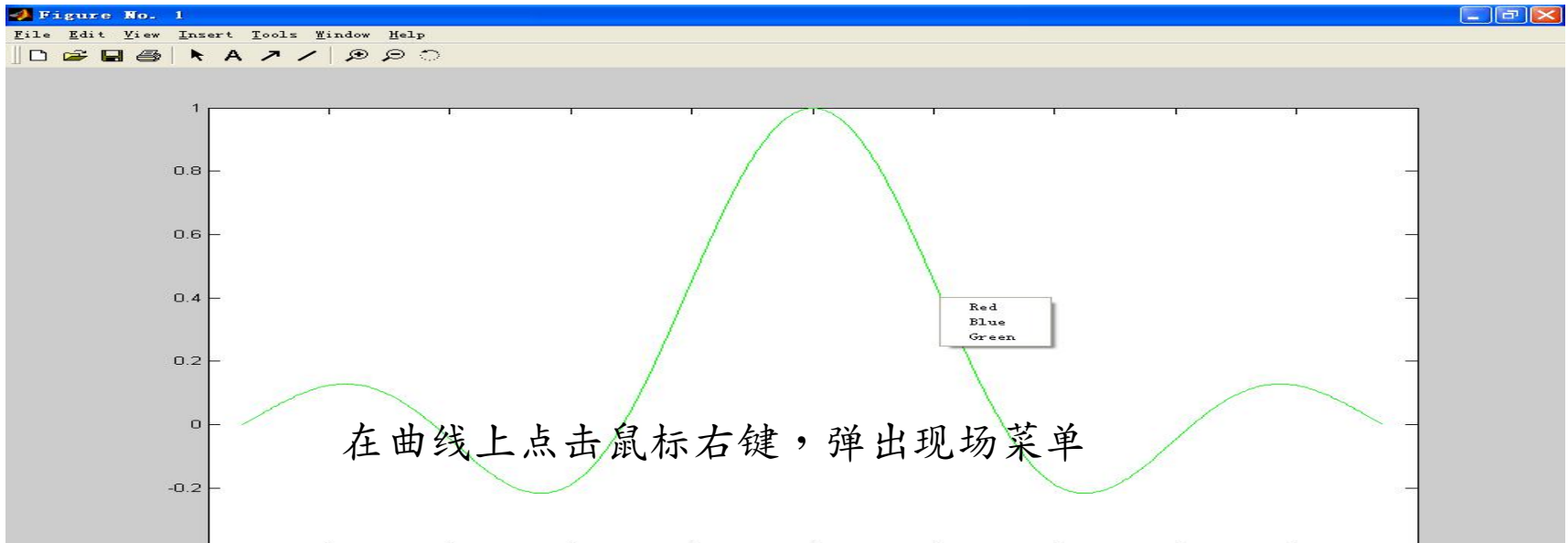
5. 快捷菜单

快捷菜单是用鼠标右键单击某对象时在屏幕上弹出的菜单。这种菜单出现的位置是不固定的，而且总是和某个图形对象相联系。在MATLAB中，可以使用uicontextmenu函数和图形对象的uiContextMenu属性来建立快捷菜单，具体步骤为：

- (1) 利用uicontextmenu函数建立快捷菜单。
- (2) 利用uimenu函数为快捷菜单建立菜单项。
- (3) 利用set函数将该快捷菜单和某图形对象联系起来。

例 绘制一条曲线 $y=\sin(t)/t$ ，创建一个与之相联系的
现场菜单，用以控制曲线的颜色

```
t=(-3*pi:pi/50:3*pi)+eps;  
y=sin(t)./t;  
hline=plot(t,y);%绘制曲线  
cm=uicontextmenu;%创建现场菜单  
%制作具体菜单项，定义相应的回调  
uimenu(cm, 'label', 'Red', 'callback', 'set(hline, "color", "r"),')  
uimenu(cm, 'label', 'Blue', 'callback', 'set(hline, "color", "b"),')  
uimenu(cm, 'label', 'Green', 'callback', 'set(hline, "color", "g"),')  
set(hline, 'uicontextmenu', cm)  
%使cm现场菜单与曲线相联系
```



7.2.3 对话框设计

对话框的控件:在对话框上有各种各样的控件，利用这些控件可以实现有关控制。

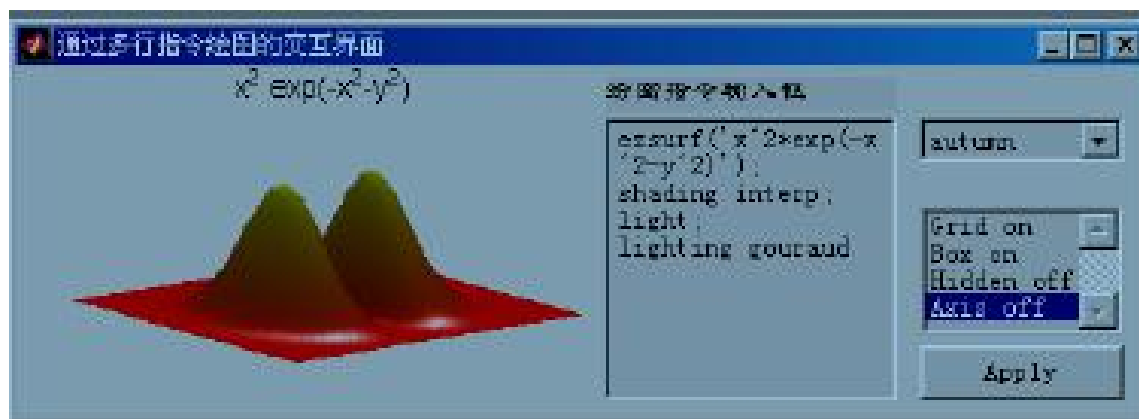
- (1) 按钮(Push Button)。
- (2) 开关按钮(Toggle Button)。
- (3) 单选按钮(Radio Button)。
- (4) 复选框(Check Box)。
- (5) 列表框(List Box)。
- (6) 弹出框(Popup Menu)。
- (7) 编辑框(Edit Box)。
- (8) 滚动条(Slider)。
- (9) 静态文本(Static Text)。
- (10) 边框(Frame)。

对话框的设计

建立控件对象的函数uicontrol，其调用格式为：

对象句柄=uicontrol(图形窗口句柄，属性名1，属性值1，属性名2，属性值2，...)

其中各个属性名及可取的值和前面介绍的uimenu函数相似



编辑框、
弹出框、
列表框、
按钮
坐标轴
静态文本框


```

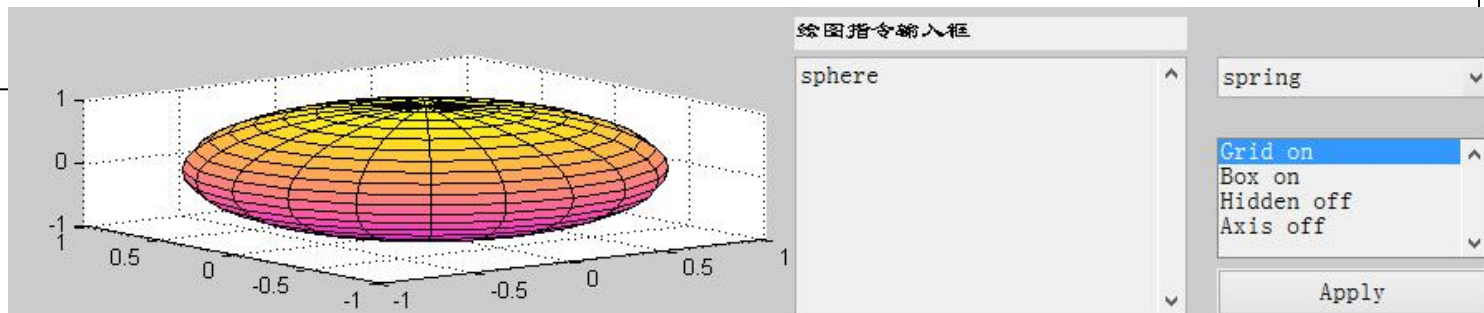
clf reset % <1>
set(gcf,'unit','normalized','position',[0.1,0.4,0.85,0.35]);%设置图形窗大小
set(gcf,'defaultuicontrolunits','normalized');
set(gcf,'defaultuicontrolfontsize',11);
set(gcf,'defaultuicontrolfontname','隶书');
set(gcf,'defaultuicontrolhorizontal','left');
set(gcf,'menubar','none'); %删除图形窗工具条
str='通过多行指令绘图的交互界面';
set(gcf,'name',str,'numbertitle','off'); %书写图形窗名
h_axes=axes('position',[0.05,0.15,0.45,0.70],'visible','off');%定义轴位框位置
uicontrol(gcf,'Style','text',... %制作静态文本框
'position',[0.52,0.87,0.26,0.1],...
'String','绘图指令输入框');
hedit=uicontrol(gcf,'Style','edit',... %制作可编辑文本框 <14>
'position',[0.52,0.05,0.26,0.8],...
'Max',2); %取2，使Max-Min>1，而允许多行输入 <16>
hpop=uicontrol(gcf,'style','popup',... %制作弹出菜单 <17>
'position',[0.8,0.73,0.18,0.12],...
'string','spring|summer|autumn|winter');%设置弹出框中选项名 <19>
hlist=uicontrol(gcf,'Style','list',... %制作列表框 <20>
'position',[0.8,0.23,0.18,0.37],...
'string','Grid on|Box on|Hidden off|Axis off',...%设置列表框中选项名 <22>
'Max',2); %取2，使Max-Min>1，而允许多项选择 <23>
hpush=uicontrol(gcf,'Style','push',... %制作与列表框联用的按钮 <24>
'position',[0.8,0.15,0.18,0.12],...
'string','Grid on|Box on|Hidden off|Axis off');

```

```

function calledit(hedit,hpop,hlist)
ct=get(hedit,'string'); %获得输入的字符串函数 <2>
vpop=get(hpop,'value'); %获得选项的位置标识 <3>
vlist=get(hlist,'value'); %获得选项位置向量 <4>
if ~isempty(ct) %可编辑框输入非空时 <5>
    eval(ct) %运行从编辑文本框送入的指令 <6>
    popstr={'spring','summer','autumn','winter'}; %弹出框色图矩阵 <7>
    liststr={'grid on','box on','hidden off','axis off'};%列表框选项内容 <8>
    invstr={'grid off','box off','hidden on','axis on'};%列表框的逆指令 <9>
    colormap(eval(popstr{vpop})) %采用弹出框所选色图 <10>
    vv=zeros(1,4);vv(vlist)=1;
    for k=1:4
        if vv(k);eval(liststr{k});else eval(invstr{k});end %按列表选项影响图形
    end
end
end

```



7.3. GUIDE创建GUI

7.3.1 图形用户界面开发环境(**GUIDE**)

7.3.2 GUIDE 提供的控件工具

7.3.3 可视化的创建图形用户的工具

7.3.4 创建图形用户界面实例

7.3.1 图形用户界面开发环境(GUIDE)

1. 图形用户界面开发环境特点

- 一套可视化的创建图形窗口的工具，通过**单击和拖拉**很容易创建GUI界面；
- GUIDE保存GUI界面时，同时生成两个文件为：一个FIG文件，一个M文件。
- FIG文件扩展名.fig,是一种存放GUI的布局及所有控件的相关信息；
- M文件扩展名为.m,存放着GUI的初始代码及相关的回调函数的模板，为实现**回调函数**提供了一个参考框架。
- 用户使用M文件编辑器可以根据这个参考框架向**回调事件中添加代码**，编制自己的应用程序。

2. GUIDE创建GUI步骤

GUI界面设计和控件编程两部分：

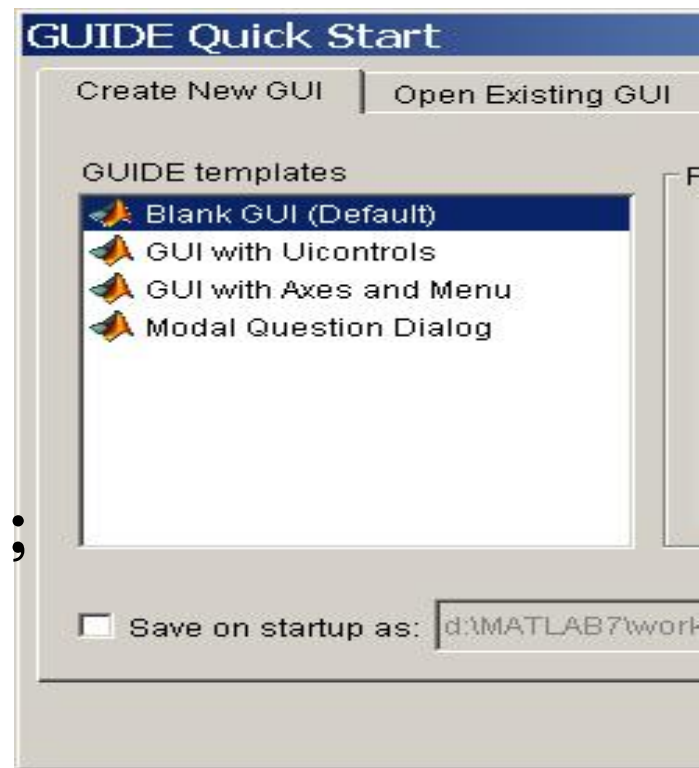
- 使用界面设计编辑器进行界面设计；
- 编写控件行为响应控制（即回调函数）代码。

3.启动和访问GUIDE模板：

- 输入命令:启动guide或访问guide filename打开如下图所示的界面；
- 菜单方式:如果Matlab已经打开,通过【File】菜单下的【New】选项中GUI也可以打开如下图所示的界面。

4. GUI的4种设计模板：

- 空白模板；
- 带有控制按钮模板；
- 带有坐标轴和菜单模板；
- 问答式对话框模板



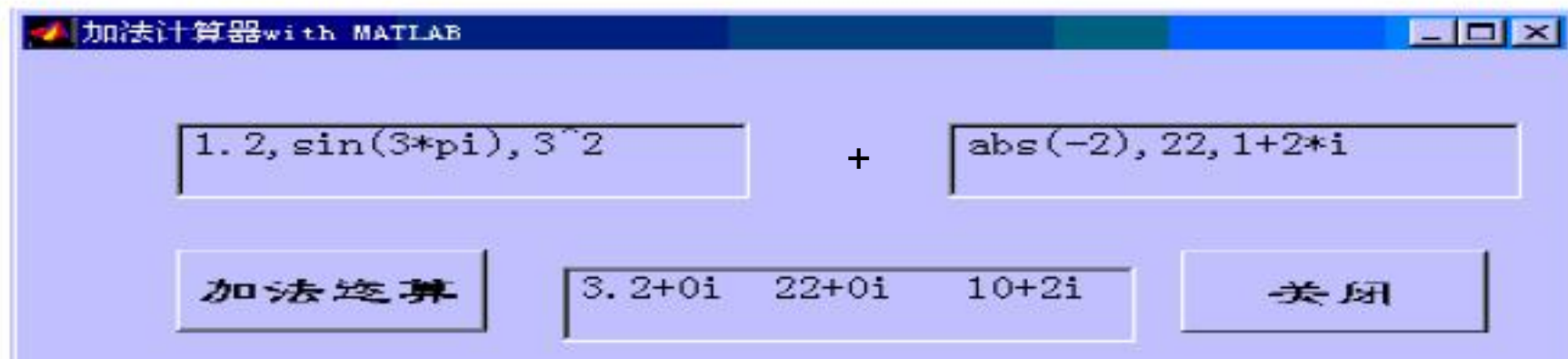
在MATLAB中，GUIDE提供多个模板来定制GUI。这些模板均已包括相关的回调函数，可以通过修改对应的M文件函数，实现指定功能。

5. 空白模板4个功能区组：



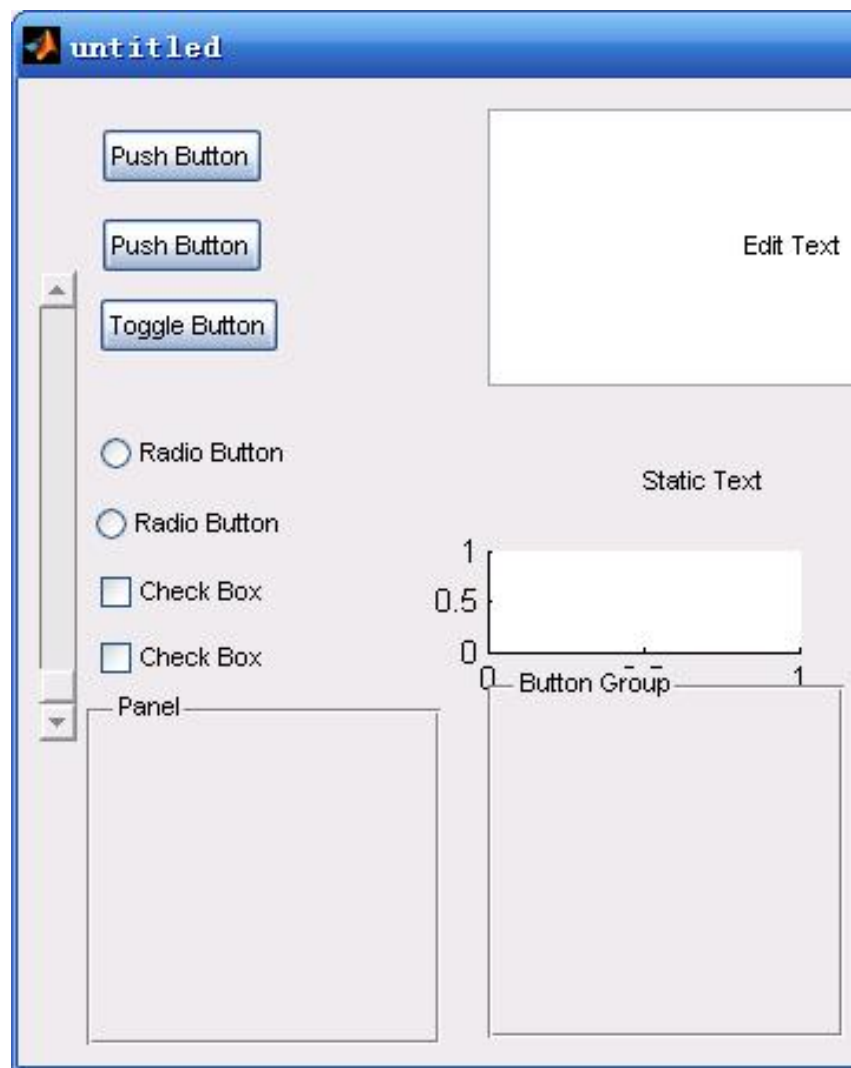
7.3.2 GUIDE 提供的控件工具





- **按钮(Push Buttons)**：通过鼠标单击按钮可以执行某种预定的功能或操作；
- **文本编辑器(Editable Texts)**：用来使用键盘输入字符串的值，可以对编辑框中的内容进行编辑、删除和替换等操作；
- **静态文本框(Static Texts)**:仅用于显示单行的说明文字.

- **滚动条(Slider)**：可输入指定范围的数量值，通过移动滚动条来改变指定范围内的数值输入，滚动条的位置代表输入数值。
- **单选按钮(Radio Button)**：单个的单选框用来在两种状态之间切换，多个单选框组成一个单选框组时，用户只能在一组状态中**选择单一**的状态，或称为单选项；
- **复选框(Check Boxes)**：单个的复选框用来在两种状态之间切换，多个复选框组成一个复选框组时，可使用户在一组状态中作组合式的选择，或称为**多选项**；

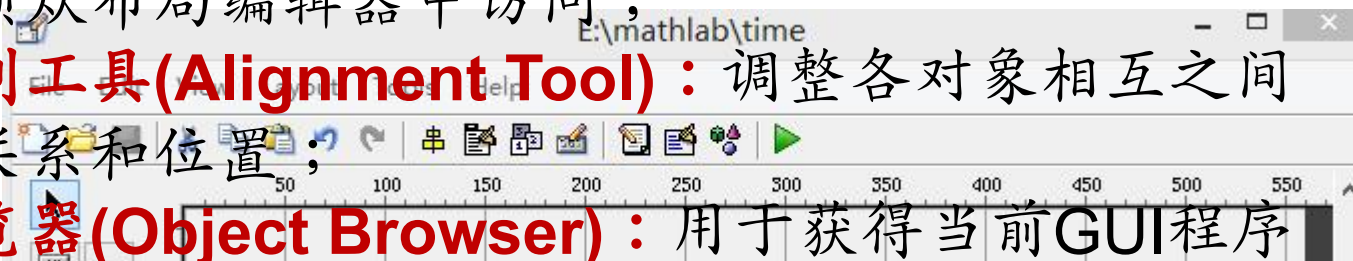


- **弹出式菜单(Popup Menus)**：让用户从一系列菜单项中选择一项作为参数输入。
- **列表框(List Boxes)**：列表框显示列表项，并能够选择其中的一项或多项。
- **开关按钮(Toggle Button)**：产生一个一个二进制状态动作（开或关），当鼠点击它时按钮将下陷，并执行callback（**回调函数**）中指定的内容，再次点击，按钮复原，并再次执行callback 中的内容。

- **组合框（面板）panel**：组合框是图形窗口中的一个封闭区域，它把相关联的控件组合在一起。
- **按钮组button group**：按钮组类似于组合框，但是它可以响应单选按钮以及开关按钮的高级属性。
- **坐标轴axes**：坐标轴可以设置关于外观和行为的参数。

7.3.3 可视化的创建图形用户的工具

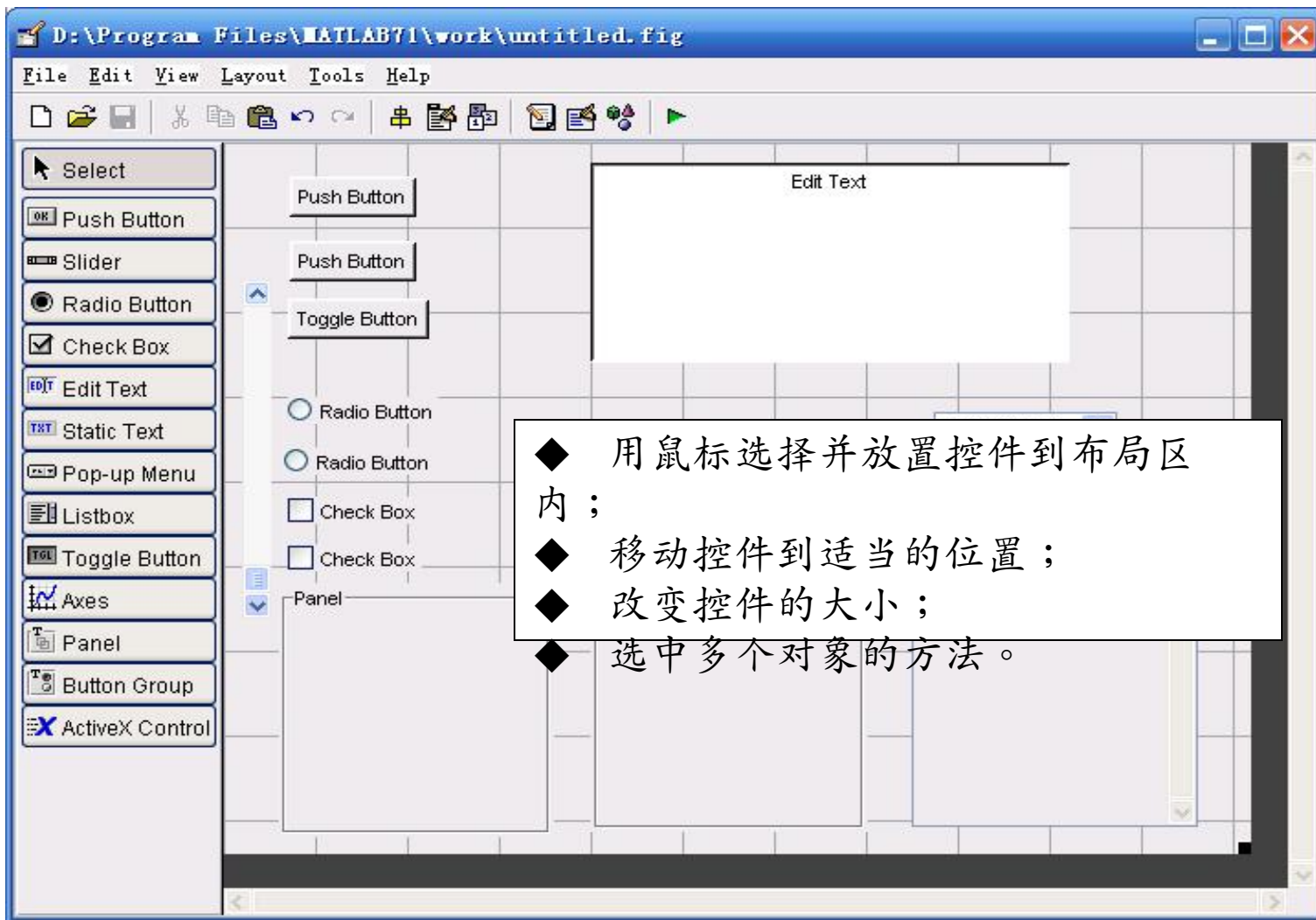
1. **界面布局编辑器(Layout Editor)**：在图形窗口中创建及布置图形对象。可以启动用户界面的控制面板，上述工具都必须从布局编辑器中访问；
2. **几何排列工具(Alignment Tool)**：调整各对象相互之间的几何关系和位置；
3. **对象浏览器(Object Browser)**：用于获得当前GUI程序中全部对象信息、类型，同时显示控件的名称和标识，在控件上双击鼠标可以打开该控件的属性编辑器；
4. **Tab顺序编辑器 (Tab Order Editor)**：用于设置当用户按下键盘上的Tab键时，对象被选中的先后顺序。
5. **属性查看器(Property Inspector)**：查询并设置属性值
6. **菜单编辑器(Menu Editor)**：创建、设计、修改下拉式菜单和快捷菜单；
7. **工具栏编辑器 (toolbar layout)**：创建、设计、修改工具栏



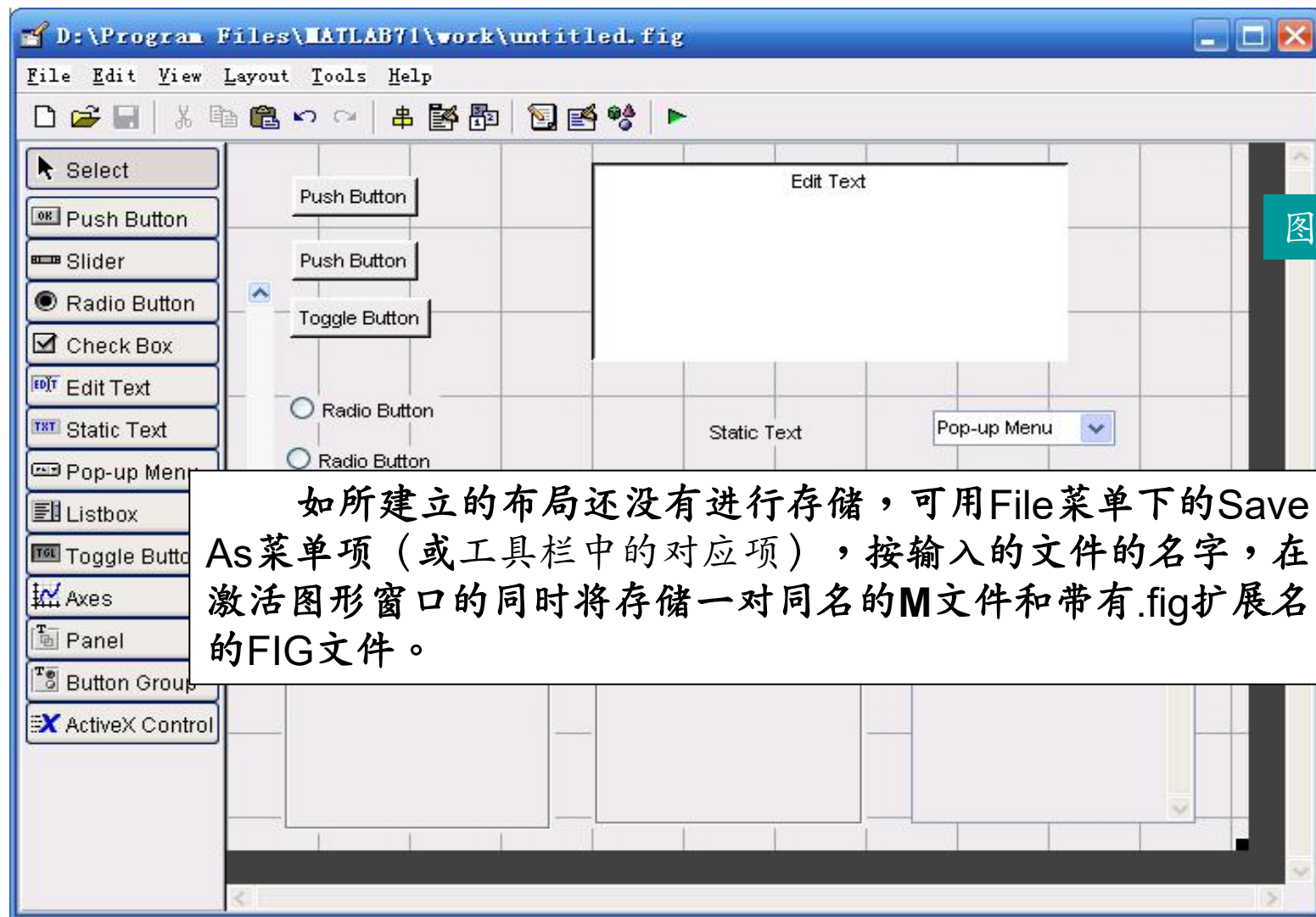
1. 布局编辑器(Layout editor)

- 或者称界面设计编辑器,用于从控件选择板上选择控件对象并放置到布局区去,布局区被激活后就成为图形窗口。
- 在命令窗口输入GUIDE命令或点击工具栏中的guide图标都可以打开空白的**布局编辑器**,在命令窗口输入GUIDE filename可打开一个已存在的名为filename**图形用户界面**。

① 将控件对象放置到布局区



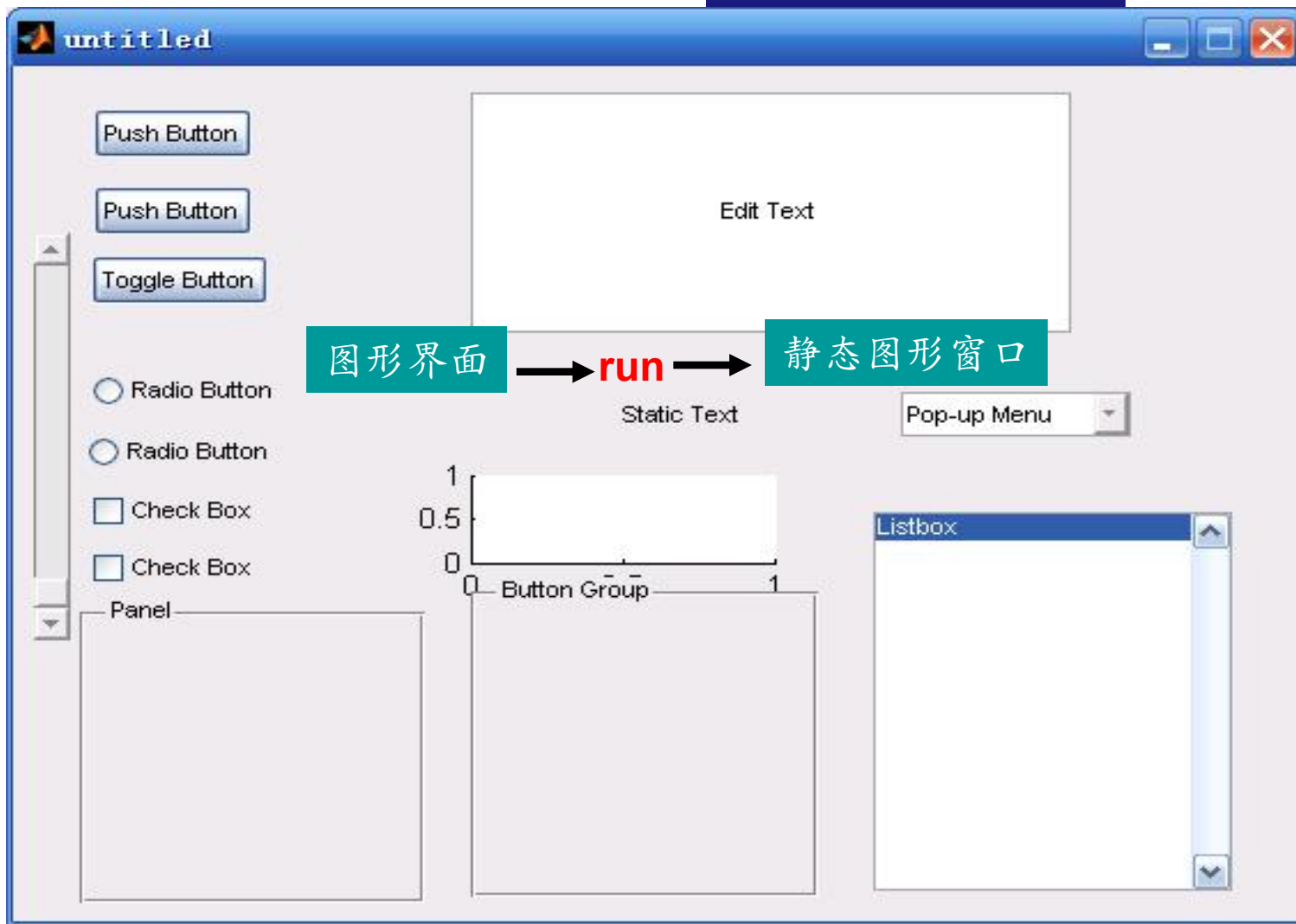
②激活图形窗口



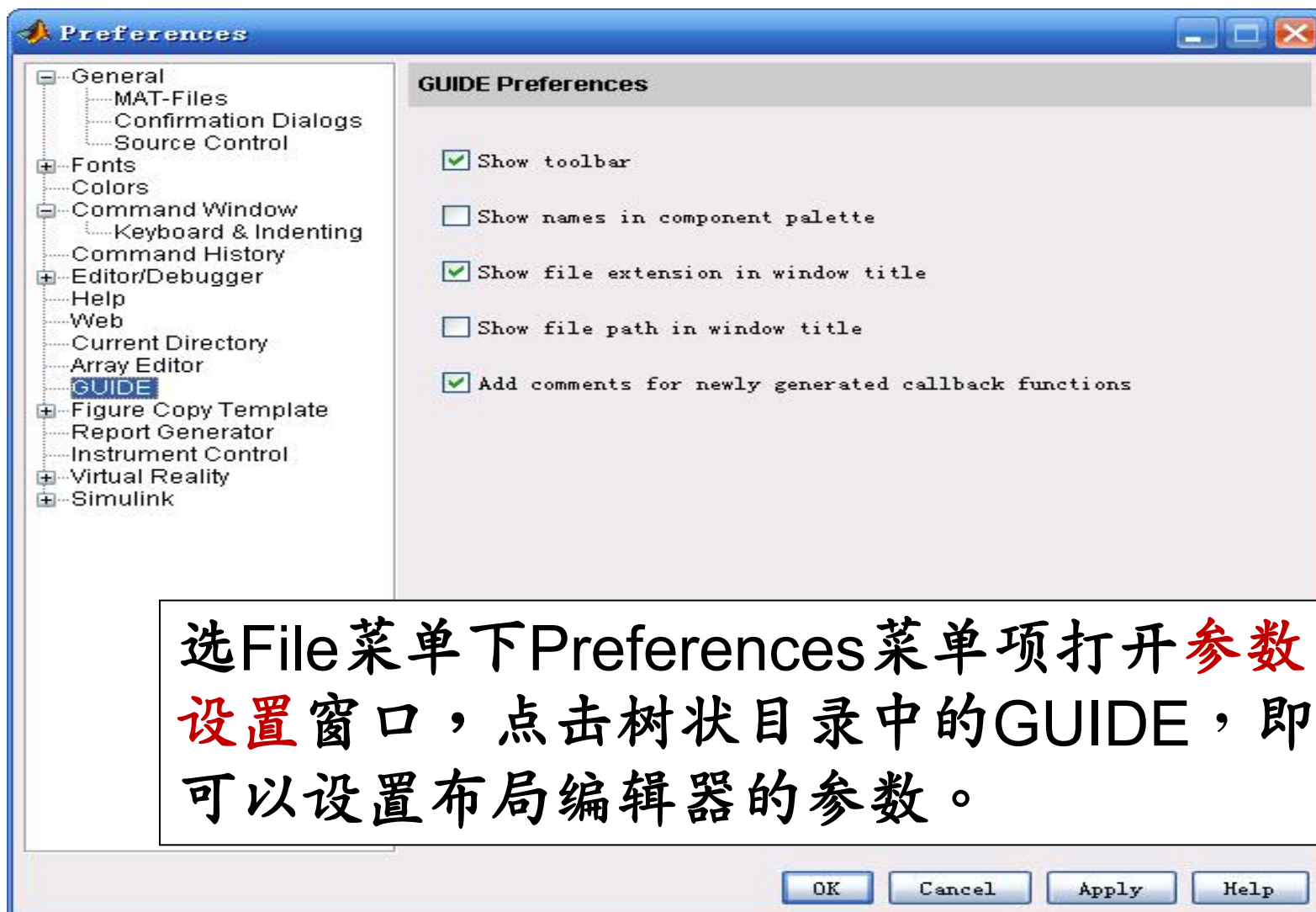
如所建立的布局还没有进行存储，可用File菜单下的Save As菜单项（或工具栏中的对应项），按输入的文件的名字，在激活图形窗口的同时将存储一对同名的M文件和带有.fig扩展名的FIG文件。

③运行GUI程序

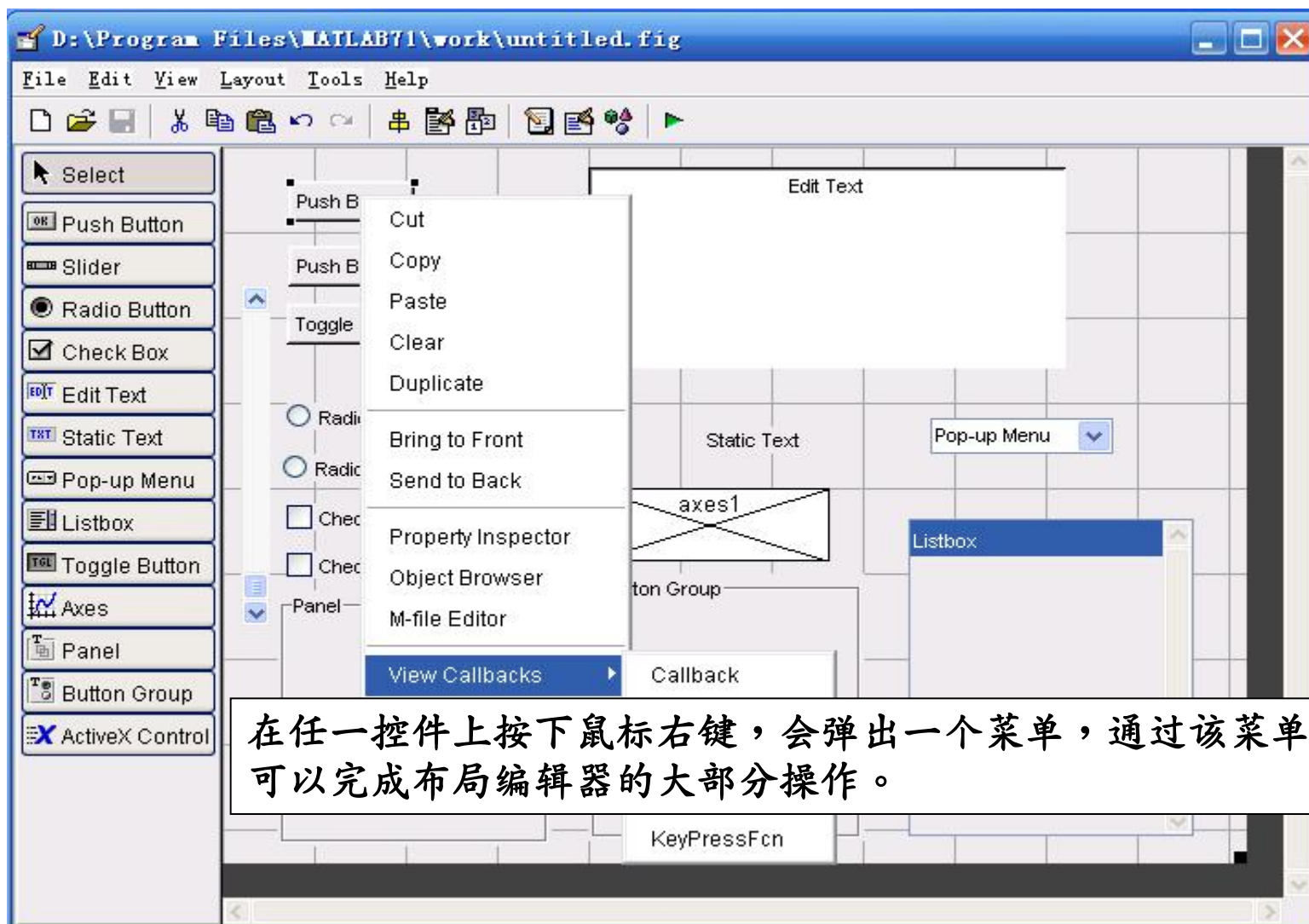
运行run



④ 布局编辑器参数设置



⑤ 布局编辑器的弹出菜单

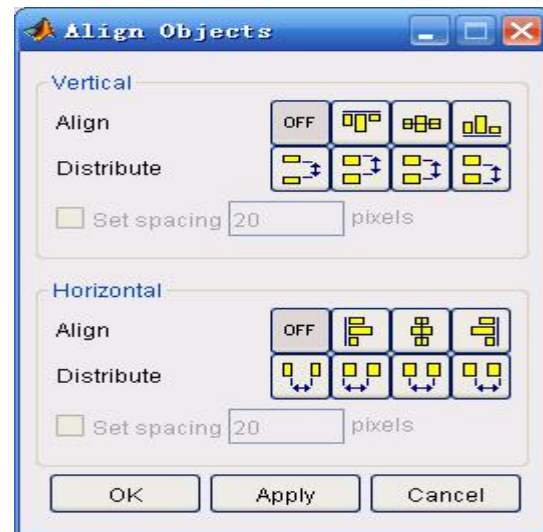


2. 位置调整工具(Alignment tool)

利用位置调整工具，可以对GUI对象设计区内的多个对象的位置进行调整。

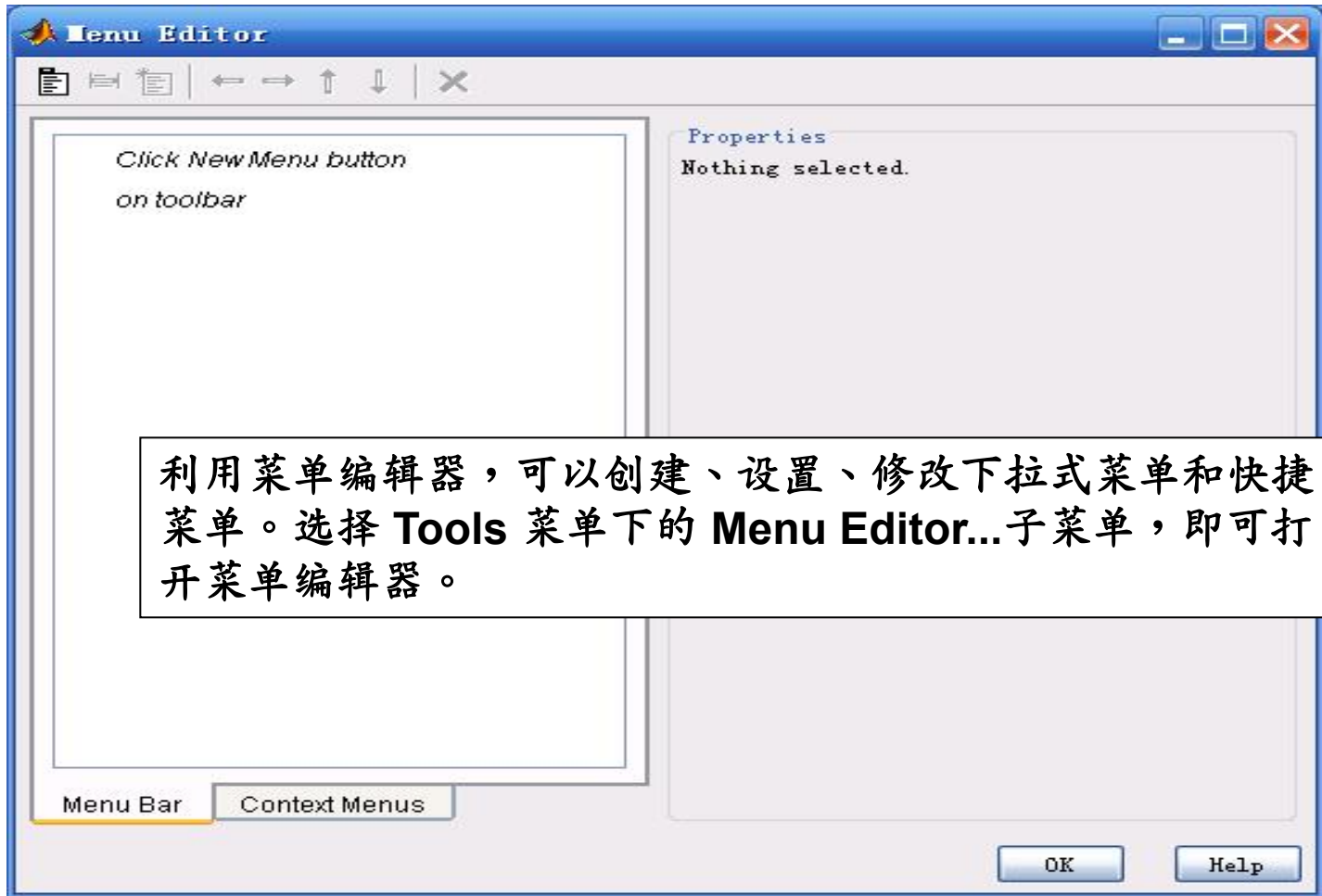
位置调整工具打开方式有两种：

- ① 从GUI设计窗口的工具栏上选择Align Objects命令按钮;
- ② 选择Tools菜单下的Align Objects...菜单项，就可以打开对象位置调整器。

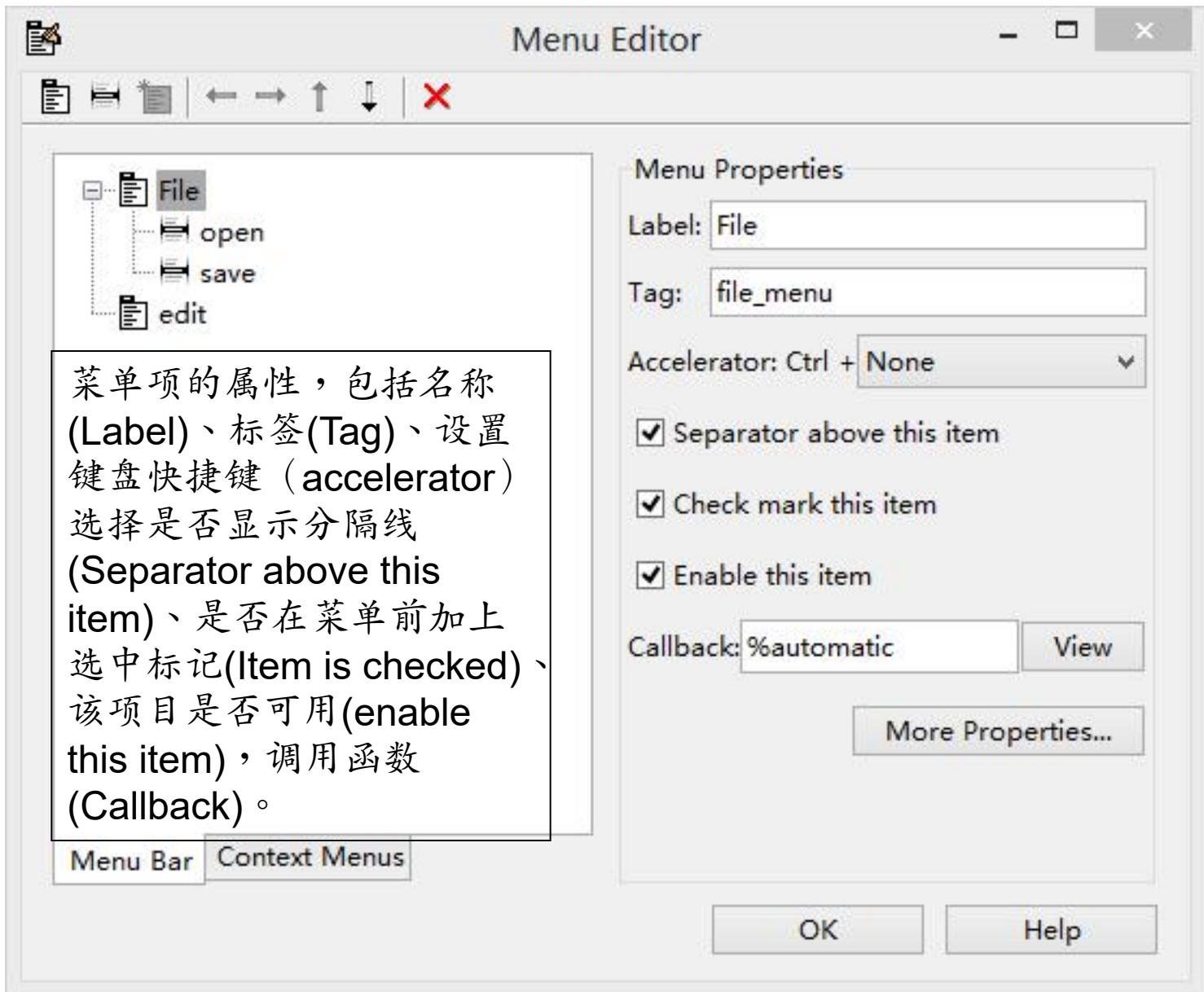


对象位置调整器中的第一栏是垂直方向的位置调整，第二栏是水平方向的位置调整。在选中多个对象后，可以方便的通过对象位置调整器调整对象间的对齐方式和距离。

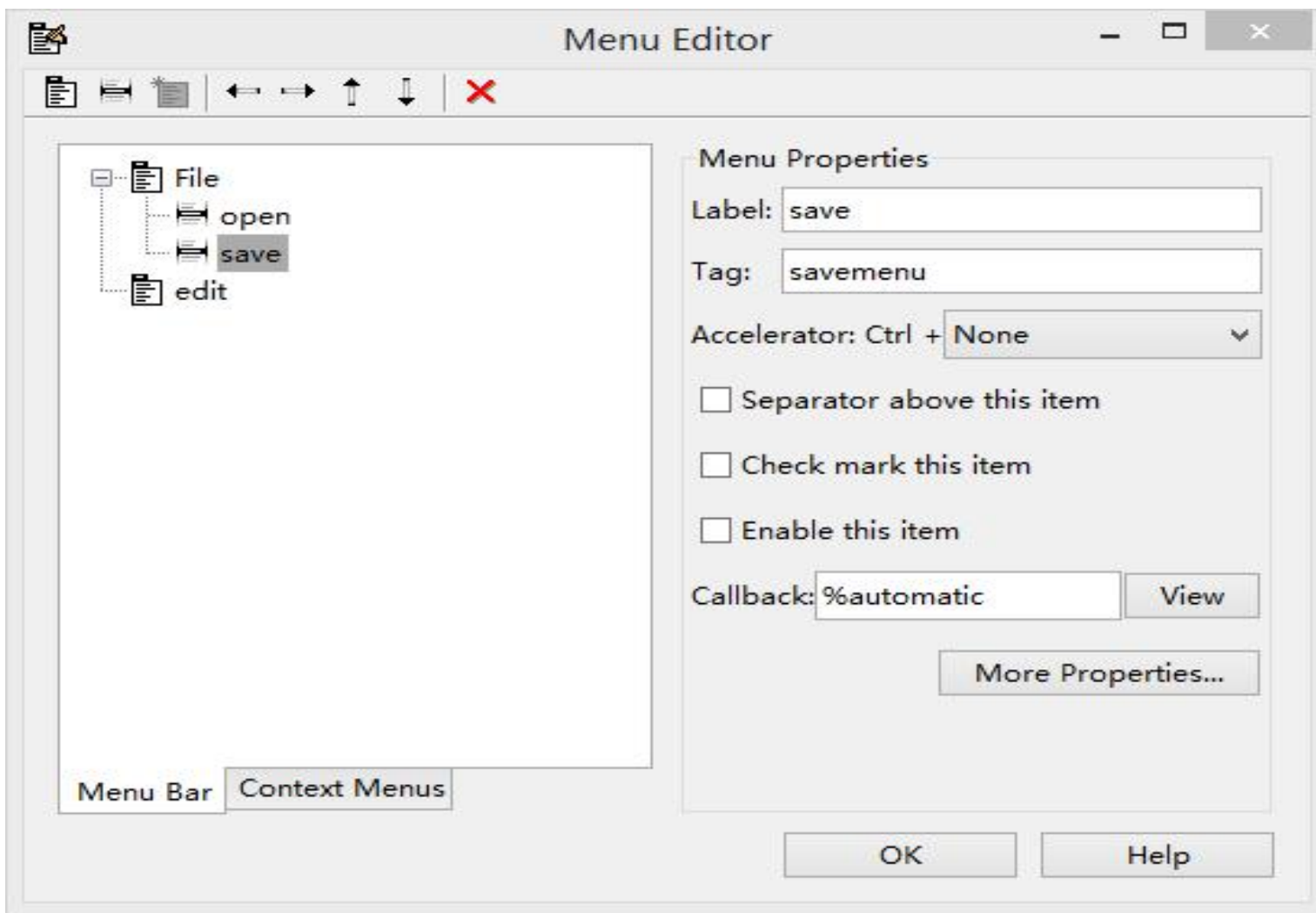
3. 菜单编辑器(Menu Editor)



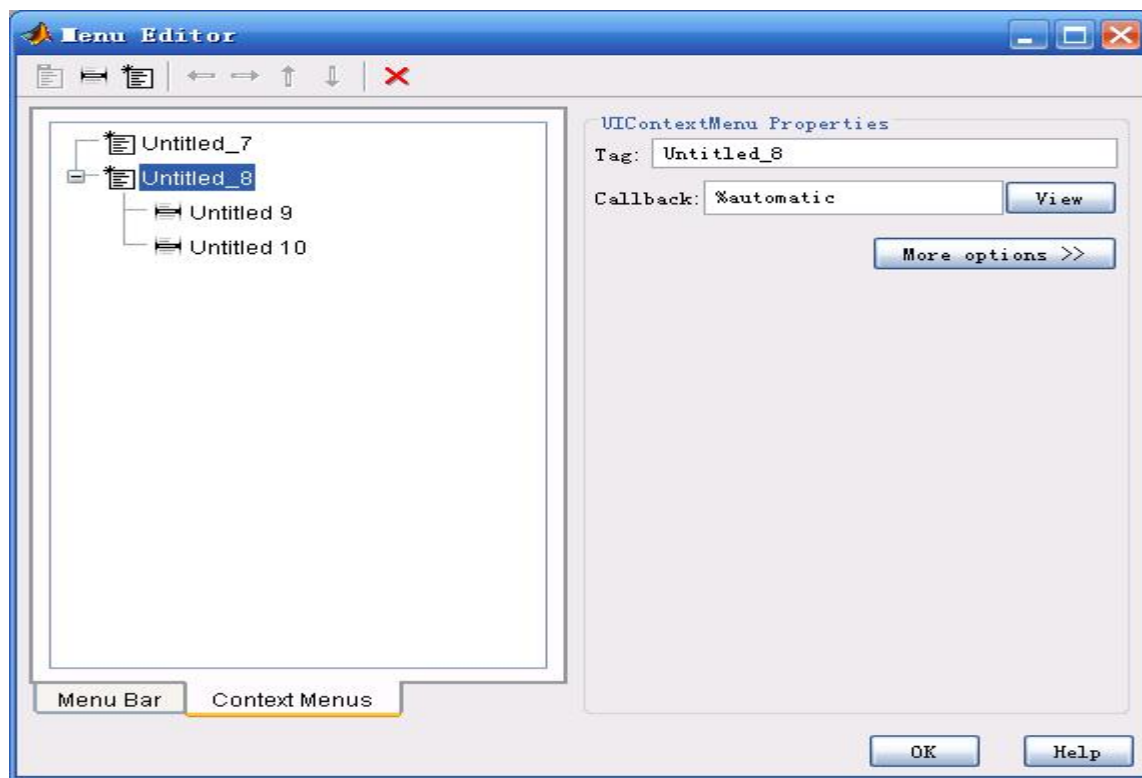
菜单编辑器有八个快捷键，可以利用它们任意添加或删除菜单，



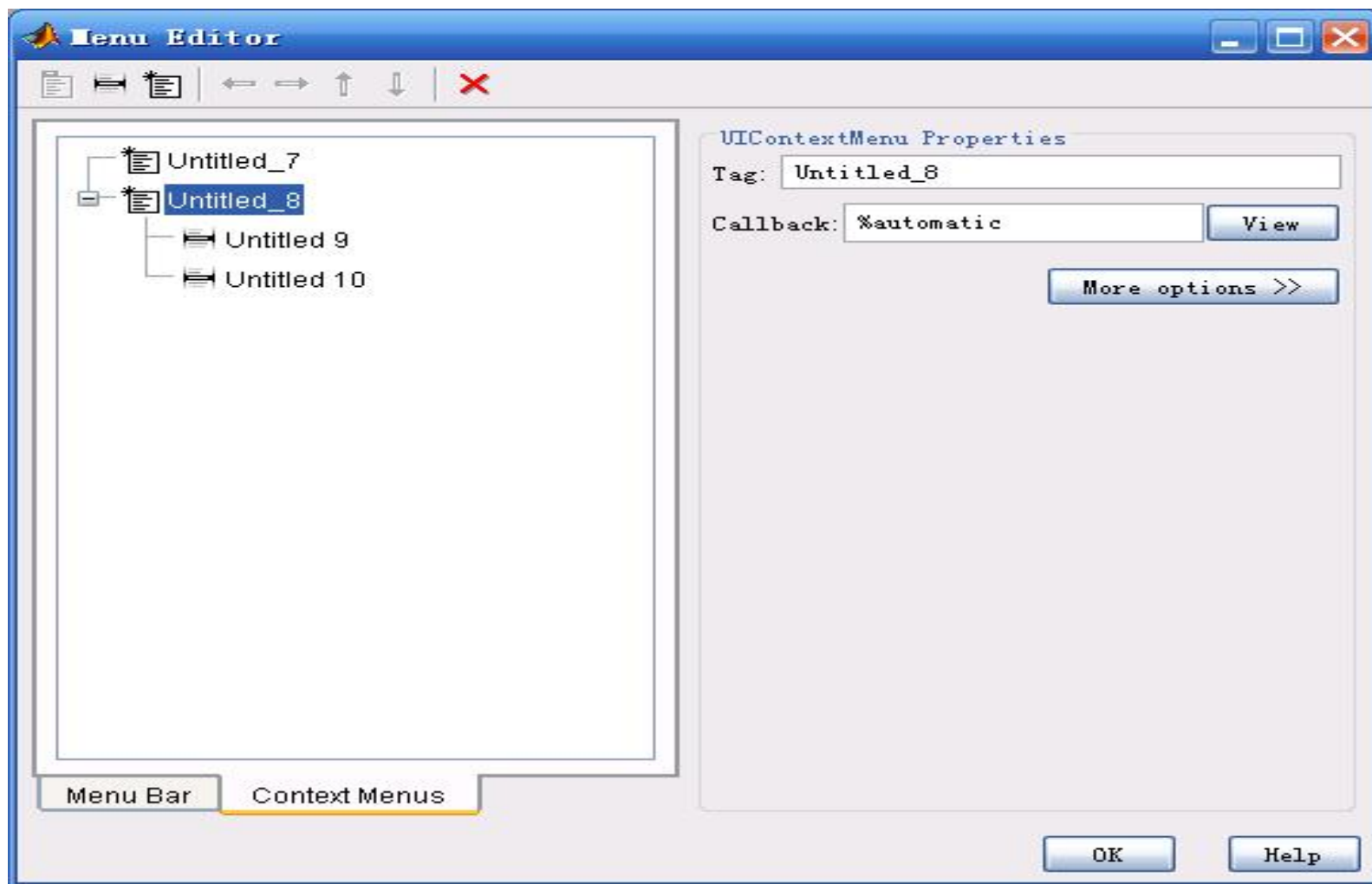
菜单编辑器左上角的第一个按钮用于创建一级菜单项。第二个按钮用于创建一级菜单的子菜单。



菜单编辑器的左下角有两个按钮，选择第一个按钮，可以创建下拉式菜单。选择第二个按钮，可以创建Context Menu菜单。选择它后，菜单编辑器左上角的第三个按钮就会变成可用，单击它就可以创建Context Menu主菜单。在选中已经创建的Context Menu主菜单后，可以单击第二个按钮创建选中的Context Menu主菜单的子菜单。与下拉式菜单一样，选中创建的某个Context Menu菜单，菜单编辑器的右边就会显示该菜单的有关属性，可以在这里设置、修改菜单的属性。



菜单编辑器左上角的第四个与第五个按钮用于对选中的菜单进行左移与右移，第六与第七个按钮用于对选中的菜单进行上移与下移，最右边的按钮用于删除选中的菜单。



4. 对象浏览器(Object Browsers)

利用对象浏览器，可以查看当前设计阶段的各个句柄图形对象。可以在对象浏览器中选中一个或多个控件来打开该控件的属性编辑器。



对象浏览器的打开方式：

- ① 从GUI设计窗口的工具栏上选择Object Browser命令按钮;
- ② 选择View菜单下的Object Browser子菜单;
- ③ 在设计区域单击鼠标右键，选择弹出菜单的 Object Browser。

5.Tab顺序编辑器 (Tab Order Editor)

用Tab顺序编辑器，设置用户按键盘上的Tab键时，对象被选中的先后顺序。



6. 用属性查看器设置控件属性

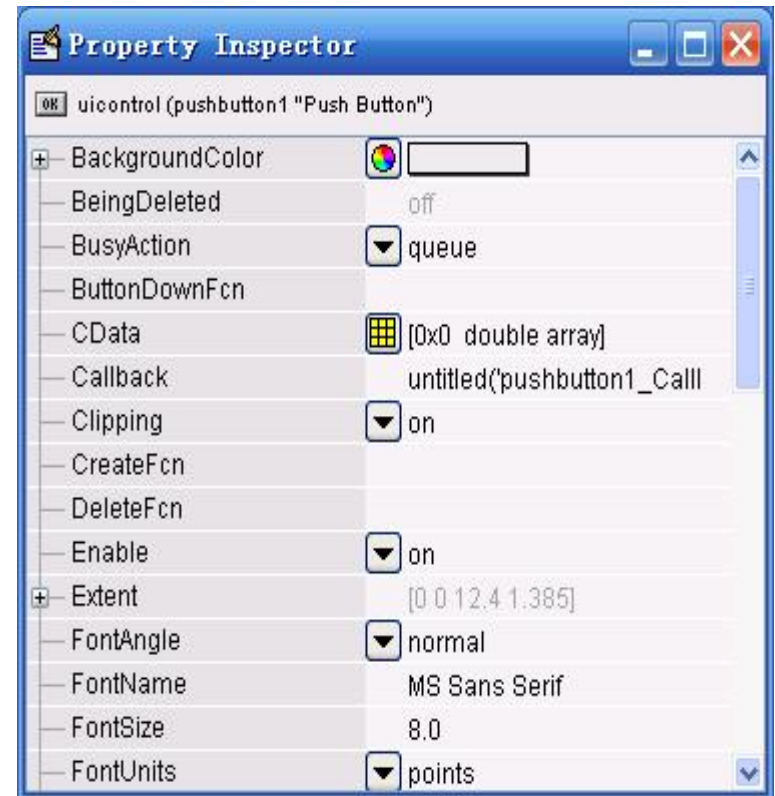
① 打开属性查看器(Opening Property Inspector)

◆从GUI设计窗口工具栏上选择Property Inspector命令按钮;

◆选择View菜单下的Property Inspector菜单项;

◆在命令窗口中输入inspect;

◆在控件对象上单击鼠标右键, 选择弹出菜单的Property Inspector菜单项。



② 使用属性查看器(Using Property Inspector)



③控件对象的基本控制属性

- ◆Resize on: 决定图形窗口缩放；
- ◆Enable：表示此控件的使能状态，设置为on”，表示可选，为“off”时则表示不可选，灰色；
- ◆Visible：控件是否可见；
- ◆BackgroundColor：定义背景颜色，取值为颜色的RGB数值；缺省值为浅灰色；
- ◆ForegroundColor：该属性定义控件对象**标题字符的颜色**；取值为颜色的预定义字符或RGB数值，缺省值为黑色；
- ◆Extend：记录控件对象**标题字符的位置和尺寸**；取值为四元素矢量[0, 0, width, height]。
- ◆Max，Min：控制件的大小，取值都为数值，缺省值分别为1和0。

❑String：定义**控件对象标题**或选项内容；取值为字符串矩阵或块数组，

❑Tag：控件标示符，回调函数名称由**Tag+callback**

❑Style：控件对象类型；取值可以是pushbutton(缺省值), radiobutton, checkbox, edit, text, slider, frame, popupmenu 或listbox；

❑Units：设置控件的位置及大小的单位，缩放时保持该区比例；影响着一切**定义大小的属性相**，取值可以是pixels (像素，缺省值), normalized (相对单位), inches, centimeters (厘米) 或points (磅)；

❑Value：控制件的当前值；取值可以是矢量，也可以是数值，其含义及解释依赖于**控件对象的类型**。

④ 控件对象的修饰控制属性

- ◆ FontAngle：取值为normal（正体，缺省值），italic（斜体），oblique（方头）；
- ◆ FontName：取值为控件标题等字体的字库名；
- ◆ FontSize：取值为数值；
- ◆ FontUnits：定义字号单位，缩放时保持字体比例，取值为points（缺省值），normalized, inches, centimeters或pixels；
- ◆ FontWeight：定义字符的粗细；取值为normal（缺省值），light，demi和bold，HorizontalAligment：取值为left，center缺省值)或right，定义控件对象标题等的对齐方式。

⑤ 控件回调函数

CallBack属性：是连接程序界面整个程序系统的实质性功能的纽带。CallBack为一般回调函数，因不同的控件而已异。例如按钮被按下时发生，下拉框改变值时发生，sliderbar 拖动时发生等等。

CreateFcn：fcn是function缩写，对象产生过程中执行回调函数，是在控件对象创建的时候发生(一般为初始化样式，颜色，初始值等)

DeleteFcn：删除对象过程中执行的回调函数。

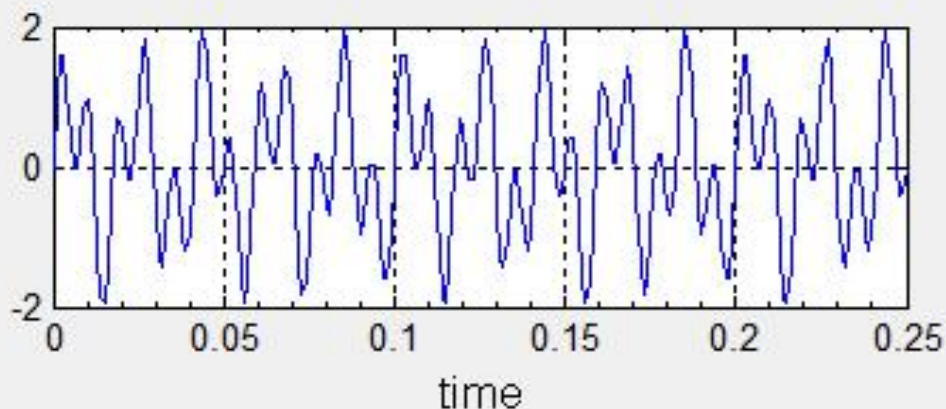
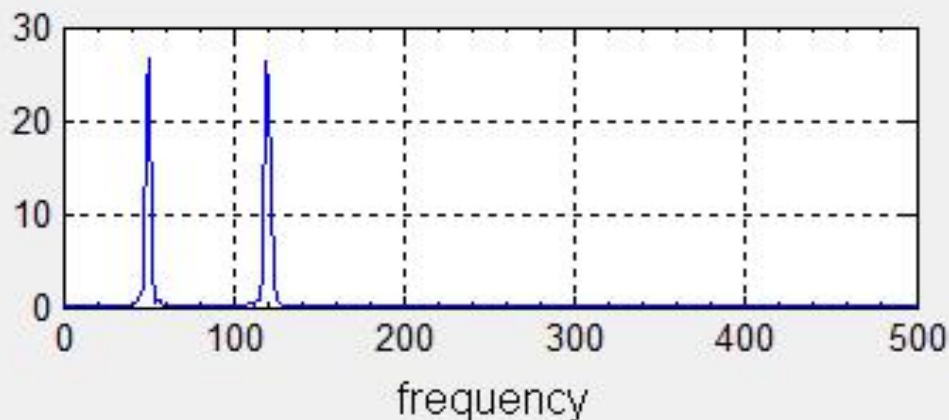
ButtonDownFcn属性：按钮按下时的处理函数。

BusyAction：处理回调函数的中断。两种选项：即Cancel：取消中断事件，queue：排队（默认设置）。

Interruptible属性：指定当前的回调函数在执行时是否允许中断，去执行其他的函数。

7.3.4 创建图形用户界面

例题：绘制 $X=\sin(2\pi f_1 t)+\sin(2\pi f_2 t)$ 和其快速傅里叶叶（FFT）图像



f1

50

f2

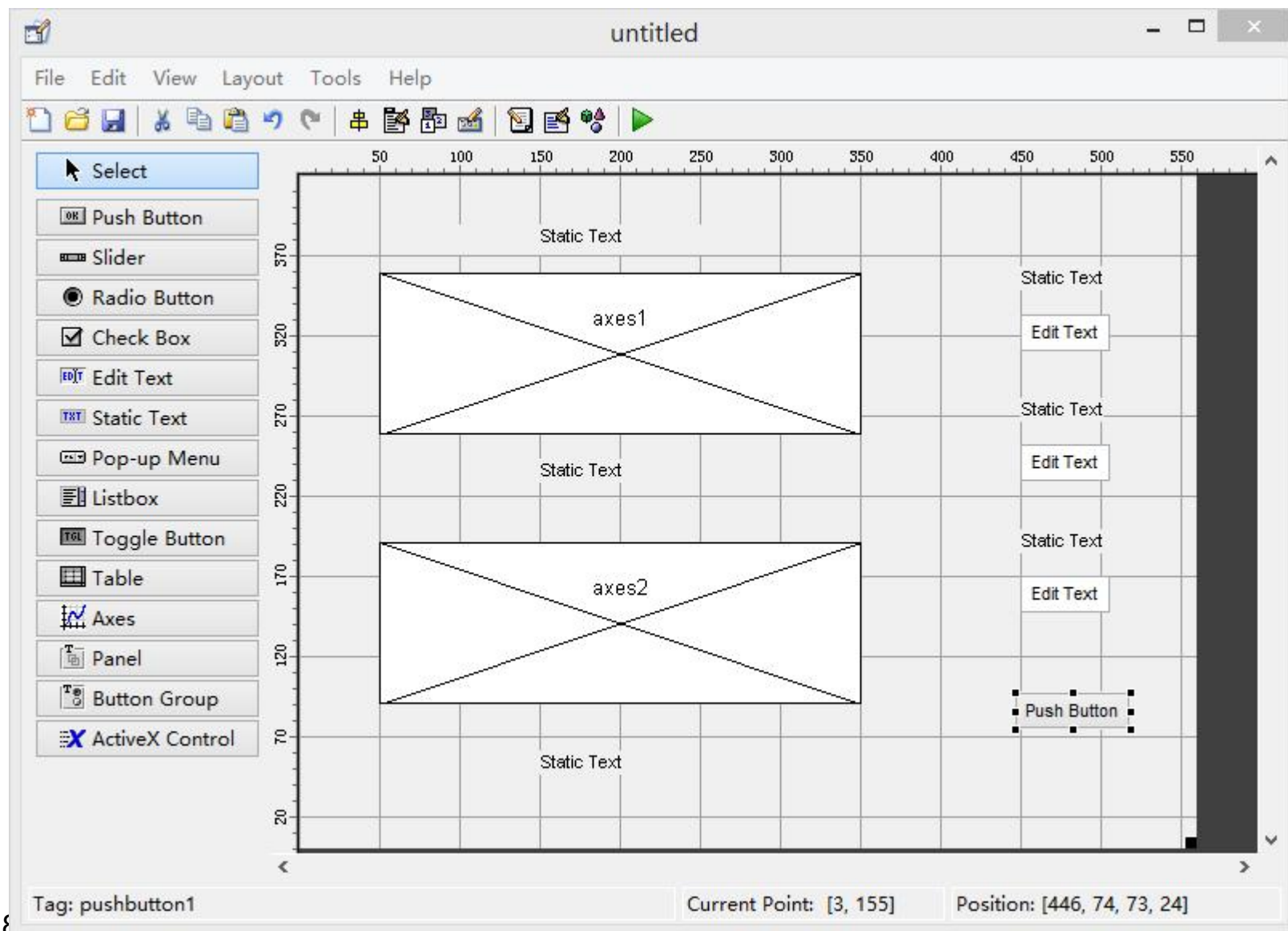
120

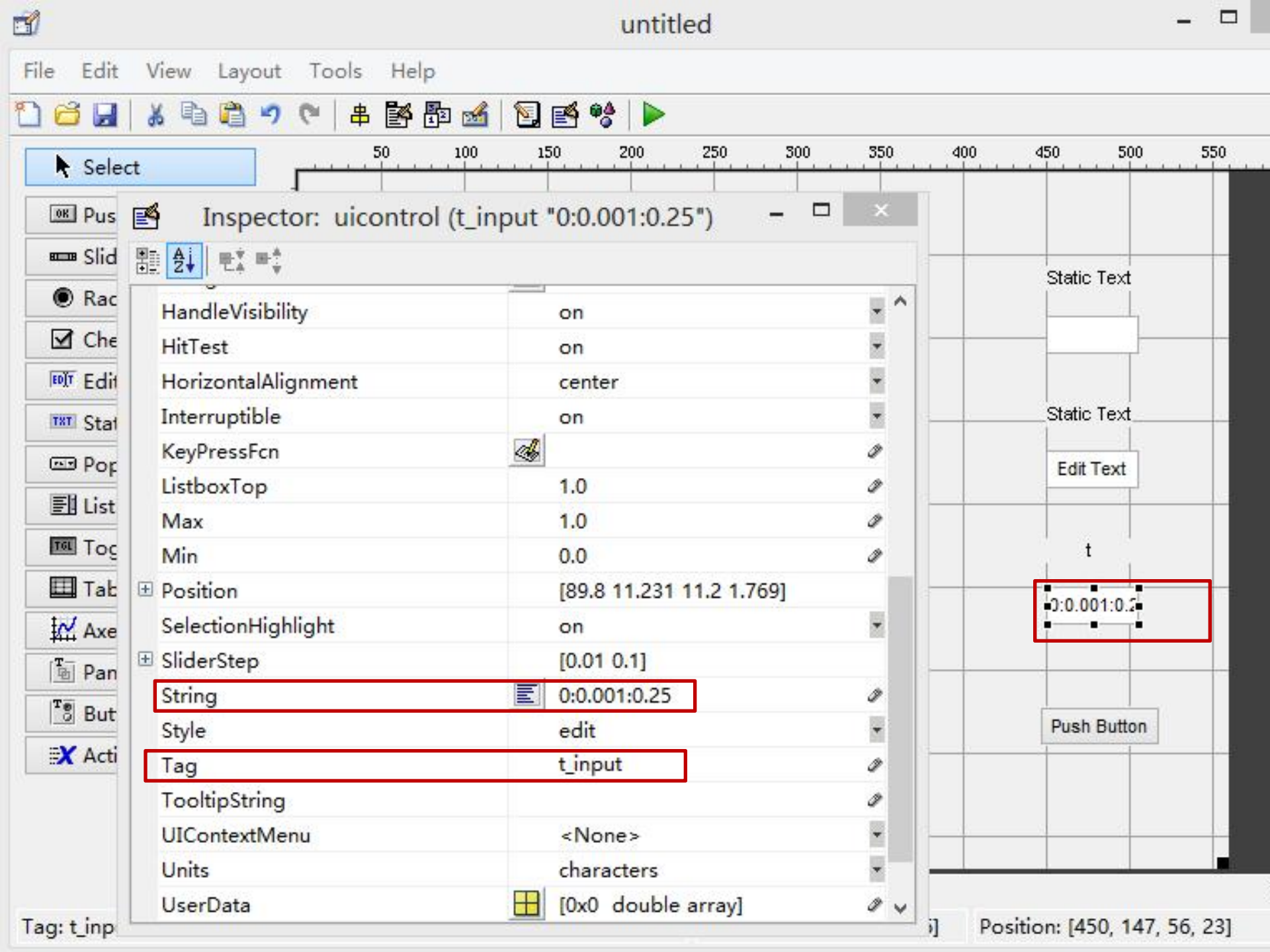
t

0:0.001:0.2

plot

创建图形用户界面





File Edit View Layout Tools Help



Select

OK Push

Slider

Radio

Checkbox

Edit

Static

Pop

List

Toggle

Table

Axes

Panel

Button

Action

Tag: t_inp

Inspector: uicontrol (t_input "0:0.001:0.25")



A2

HandleVisibility

on

HitTest

on

HorizontalAlignment

center

Interruptible

on

KeyPressFcn



ListboxTop

1.0

Max

1.0

Min

0.0

Position

[89.8 11.231 11.2 1.769]

SelectionHighlight

on

SliderStep

[0.01 0.1]

String



0:0.001:0.25

Style

edit

Tag

t_input

TooltipString

UIContextMenu

<None>

Units

characters

UserData



[0x0 double array]

Static Text

Static Text

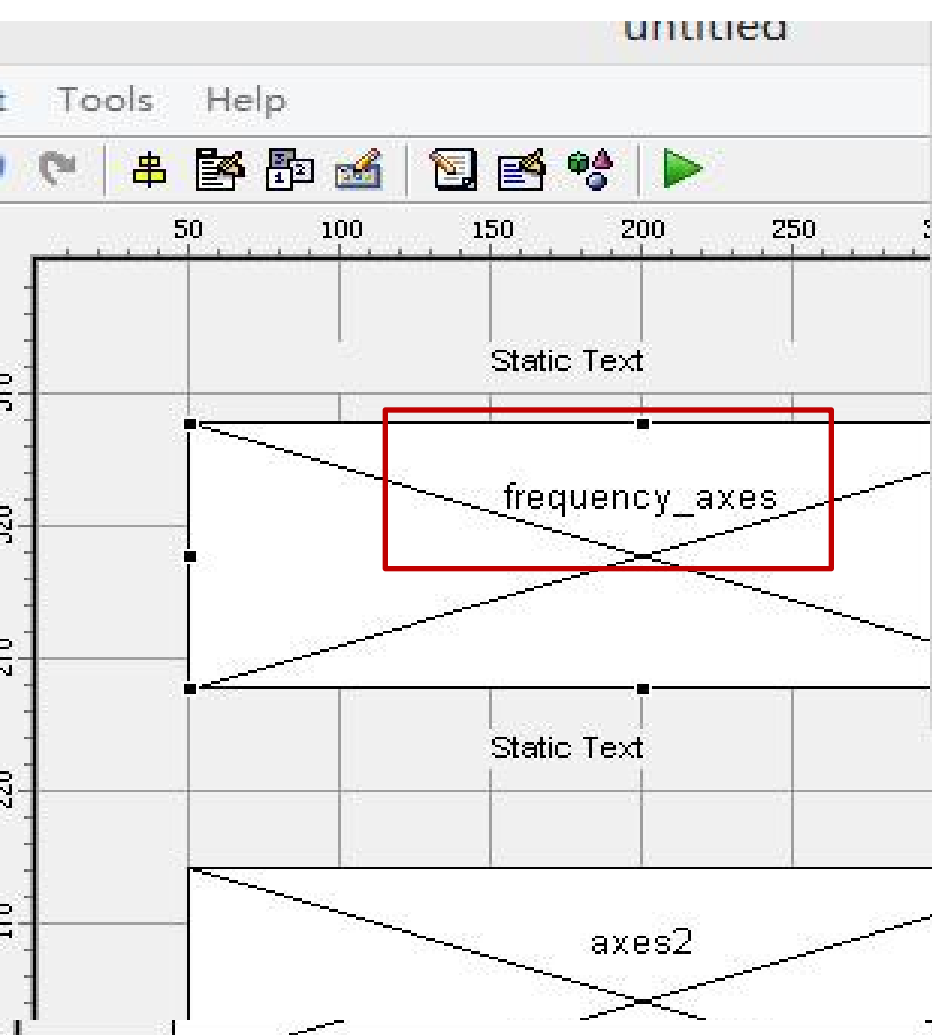
Edit Text

t

0:0.001:0.2

Push Button

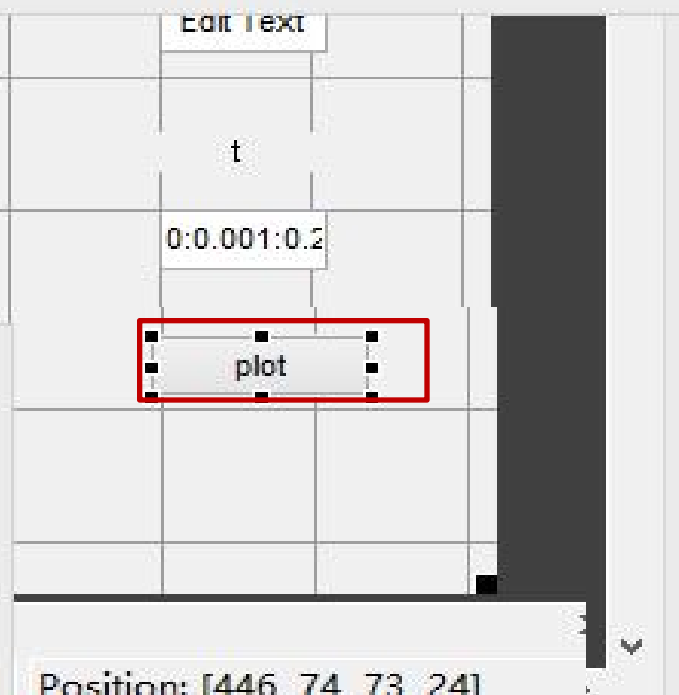
Position: [450, 147, 56, 23]



MinorGridLineStyle	:
NextPlot	replace
OuterPosition	[-4.76 16.224 85.4
PlotBoxAspectRatio	[1 1 1]
PlotBoxAspectRatioMode	auto
Position	[9.8 19.769 60.2 7
Projection	orthographic
SelectionHighlight	on
Tag	frequency_axes
TickDir	in
TickDirMode	auto

Inspector: uicontrol (plot_button "pl...

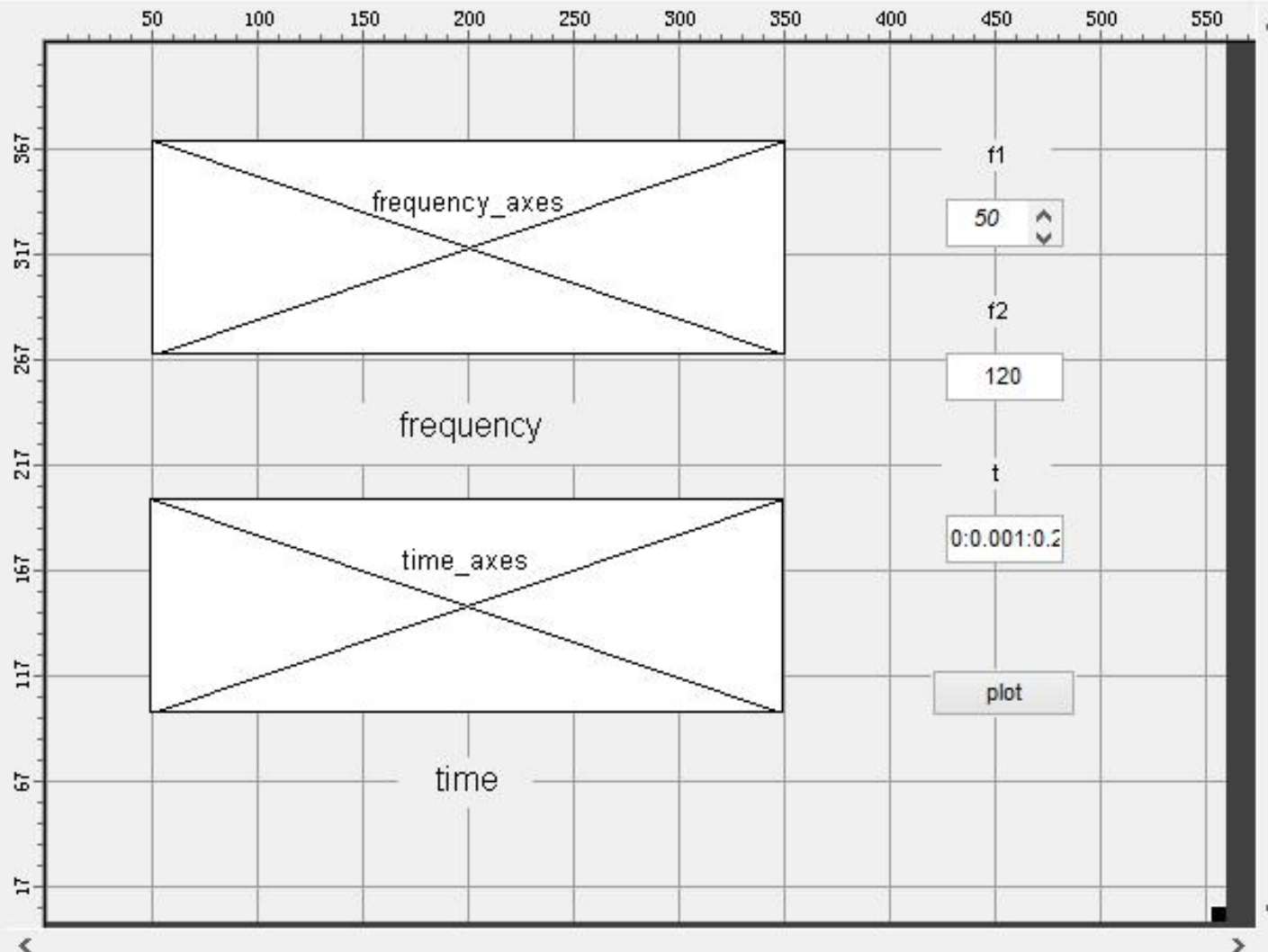
SliderStep	[0.01 0.1]
String	plot
Style	pushbutton
Tag	plot_button



File Edit View Layout Tools Help



Select

☐ Push Button☐ Slider☒ Radio Button☒ Check Box Edit Text Static Text Pop-up Menu Listbox☐ Toggle Button Table Axes Panel Button Group☒ ActiveX Control

f1

50

f2

120

t

0:0.001:0.2

plot

Tag: figure1

Current Point: [25, 344]

Position: [520, 383, 560, 417]

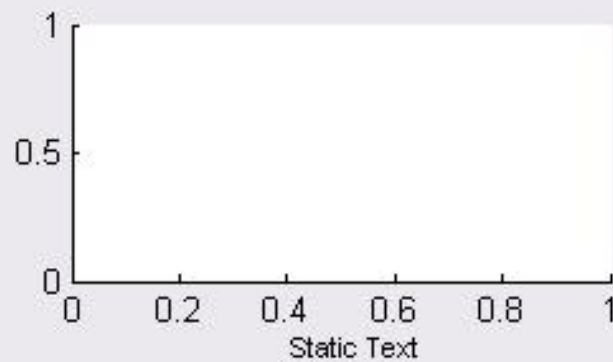
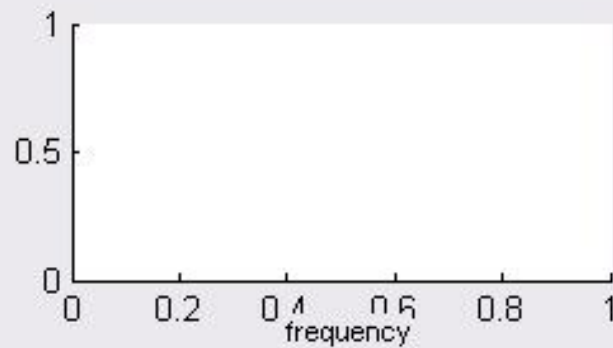
7.4 编写界面控件的回调函数

GUIDE自动生成 M文件中相应的函数体：

文件同名的主函数+各控件相应的回调子函数

- ◆ 回调函数名：控件 Tag+ Call类型名 ，
- ◆ 回调函数输入参数：hObject, eventdata, handles
- ◆ hObject：发生事件的源控件，
- ◆ eventdata：事件数据结构，保留参数，
- ◆ Handles：为一个结构体，包含图形中所有对象的句柄

Run



f1

50

f2

120

t

0:0.001:0.2

plot

静态图形窗口

文件同名的主函数

```
function varargout = time(varargin)
```

%GUI初始化函数：opening,output

```
function time_OpeningFcn(hObject, eventdata, handles, varargin)
```

```
function varargout = time_OutputFcn(hObject, eventdata, handles)
```

各控件相应的回调子函数

%GUI回调函数：

```
function f1_input_Callback(hObject, eventdata, handles)
```

```
function f1_input_CreateFcn(hObject, eventdata, handles)
```

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

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

```
function f2_input_Callback(hObject, eventdata, handles)
```

```
function f2_input_CreateFcn(hObject, eventdata, handles)
```

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

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

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

```
%get user input from GUI  
f1=str2double(get(handles.f1_input,'string'));  
f2=str2double(get(handles.f2_input,'string'));  
t=eval(get(handles.t_input,'string'));
```

```
%calculate data  
x=sin(2*pi*f1*t)+sin(2*pi*f2*t);  
y=fft(x,512);  
m=y.*conj(y)/512;  
f=1000*(0:256)/512;
```

```
%create frequency plot  
axes(handles.frequency_axes)  
plot(f,m(1:257))  
set(handles.frequency_axes,'xminor tick','on')  
grid on
```

```
%create time plot  
axes(handles.time_axes)  
plot(t,x)  
set(handles.time_axes,'xminor tick','on')
```

```
grid on
```

编写回调函数

```
%get user input from GUI  
a=str2double(get(handles.a_input,'string'));  
b=str2double(get(handles.b_input,'string'));
```

```
%calculate data  
t=-2*pi:pi/10:2*pi;  
x=a*cos(t);  
y=b*sin(t);
```

```
%create plot  
axes(handles.plot_axes)  
plot(x,y)  
grid on
```

File Edit View Layout Tools Help



time

file edit

Select

OK Push Button

Slider

Radio Button

Check Box

Edit Text

Static Text

Pop-up Menu

Listbox

Toggle Button

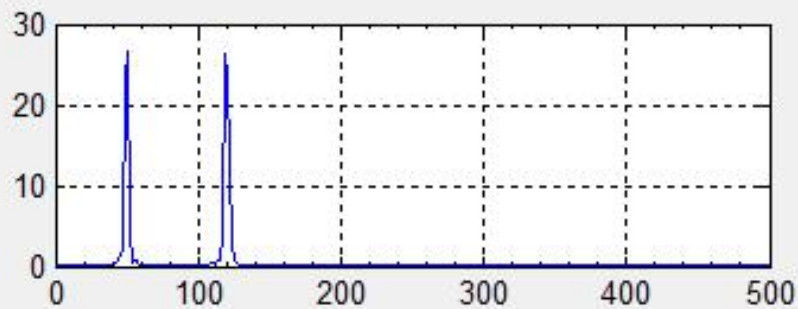
Table

Axes

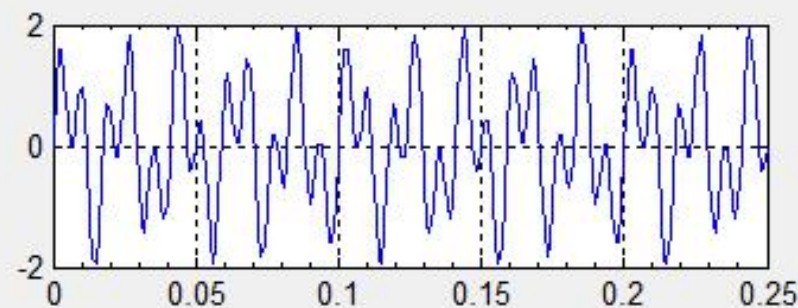
Panel

Button Group

ActiveX Control



frequency



time

f1

50

f2

120

t

0:0.001:0.2

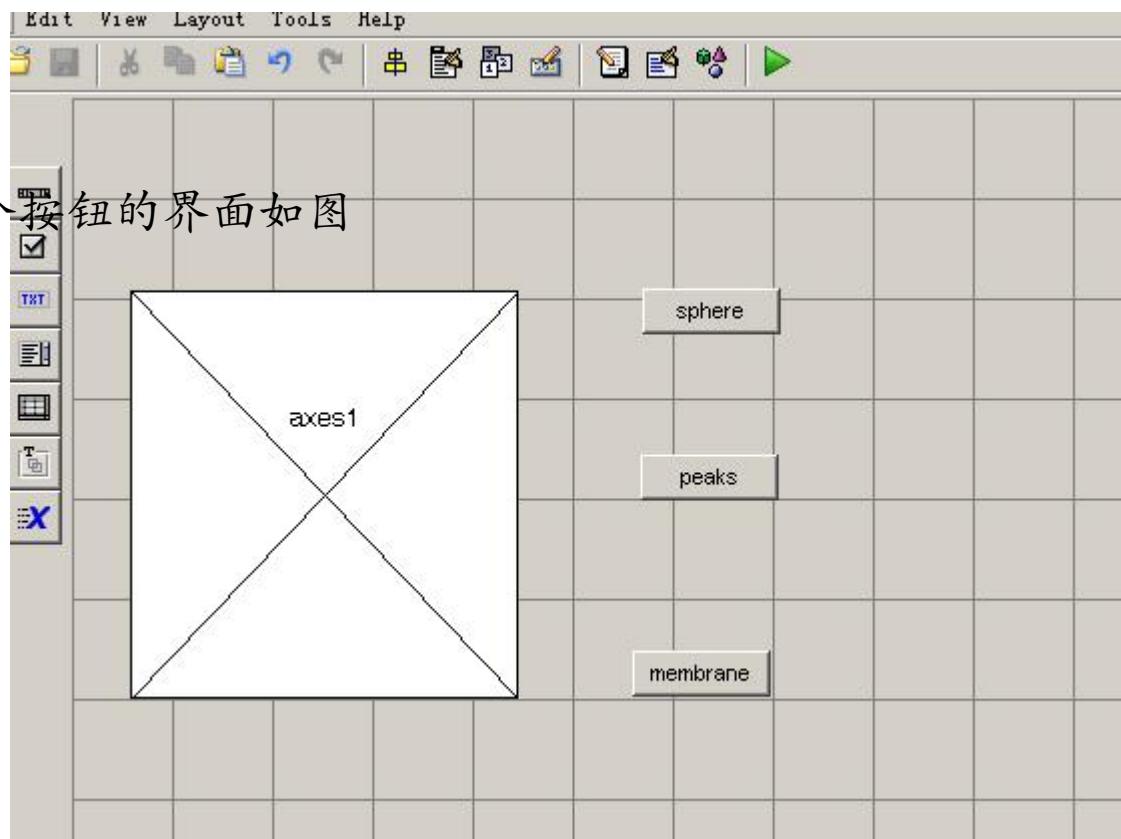
plot

图形用户界面开发环境应用示例2

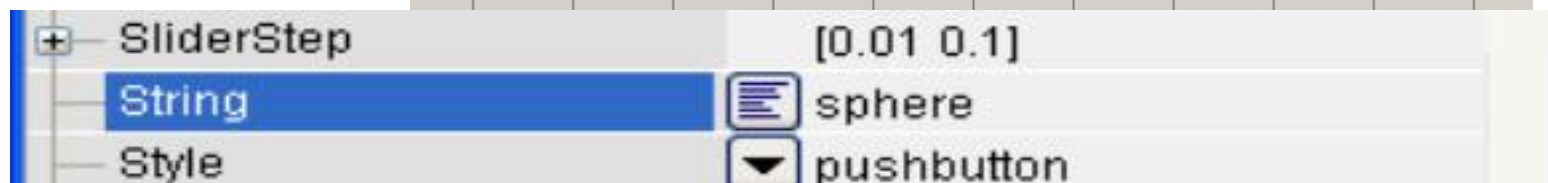
- 设计一个带有三个按钮和一个坐标轴的图形用户界面，当用鼠标点击三个按钮时，分别在坐标轴内画sphere，peaks和membrane三个图形。

➤ 创建控件

带有1个坐标轴和3个按钮的界面如图

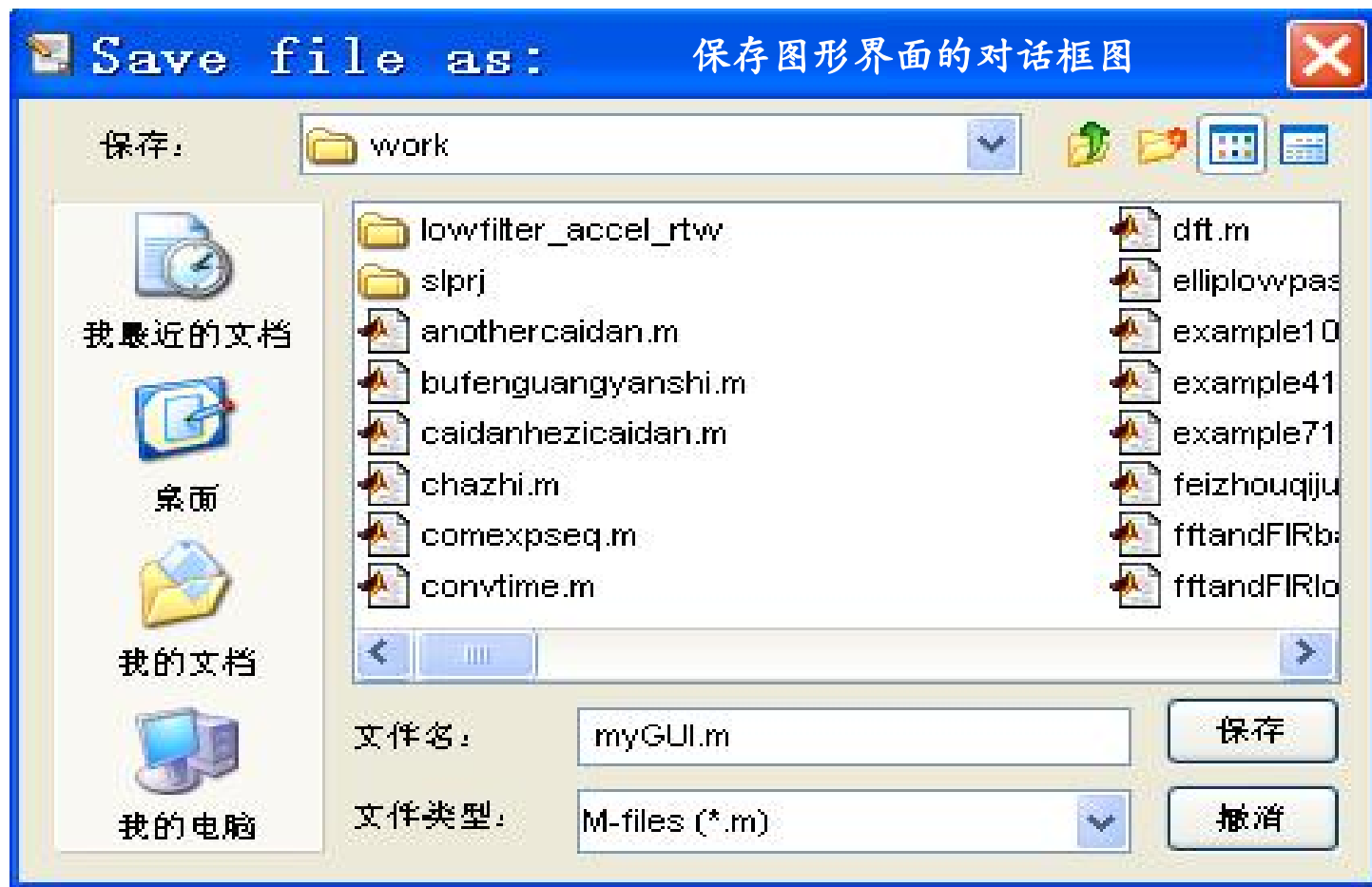


➤ 设置按钮属性

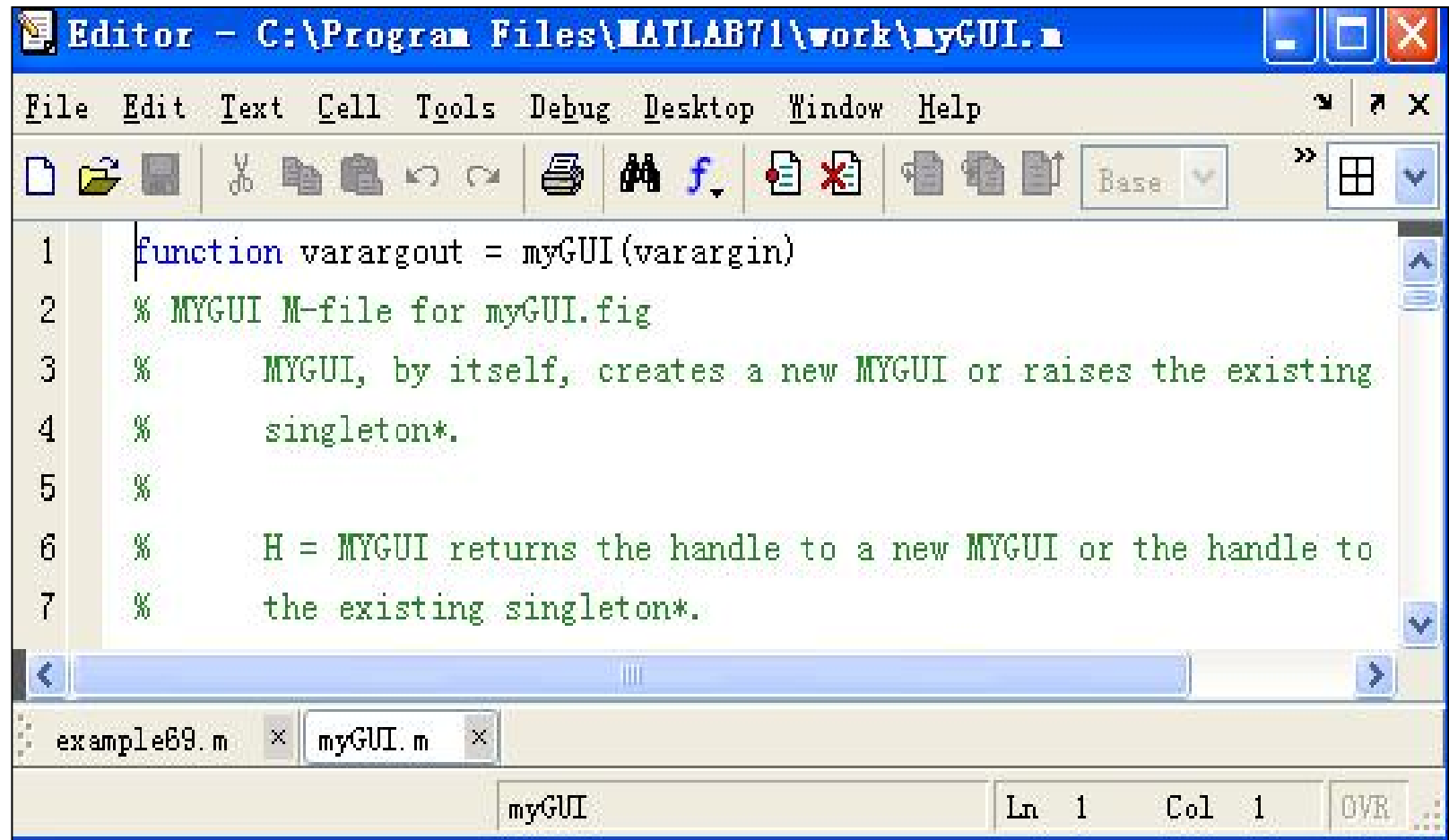


► 编写回调程序

属性设置完毕后，点击菜单栏上的保存按钮会弹出“save as”对话框，文件命名为myGUI。



自动生成的M文件

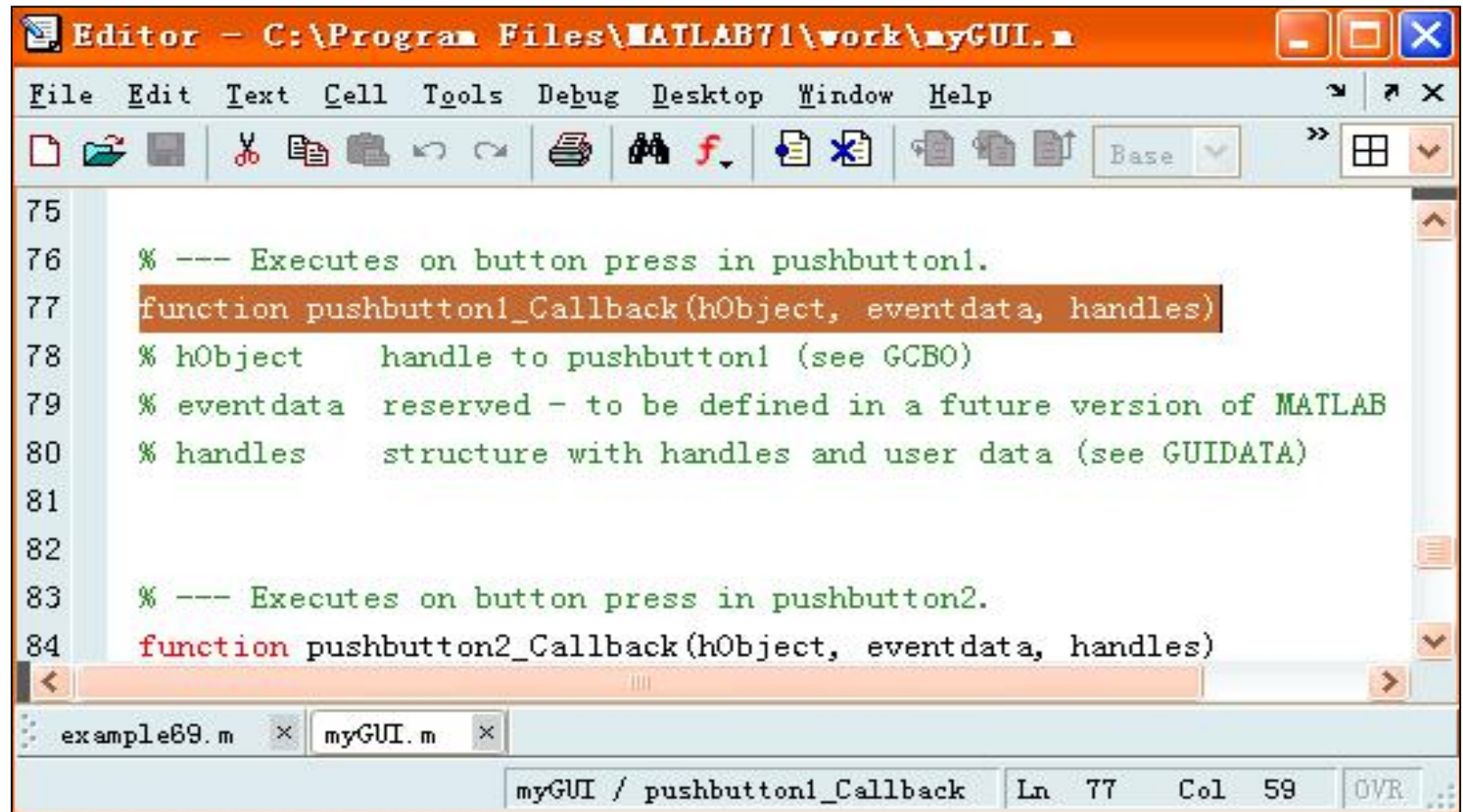


The image shows a screenshot of the MATLAB Editor window. The title bar reads "Editor - C:\Program Files\MATLAB71\work\myGUI.m". The menu bar includes "File", "Edit", "Text", "Cell", "Tools", "Debug", "Desktop", "Window", and "Help". The toolbar contains various icons for file operations, editing, and debugging. The main text area displays the following MATLAB code:

```
1 function varargout = myGUI(varargin)
2 % MYGUI M-file for myGUI.fig
3 %     MYGUI, by itself, creates a new MYGUI or raises the existing
4 %     singleton*.
5 %
6 %     H = MYGUI returns the handle to a new MYGUI or the handle to
7 %     the existing singleton*.
```

The status bar at the bottom shows "myGUI" and "Ln 1 Col 1".

按钮1的回调子函数定位图



The image shows a MATLAB Editor window titled "Editor - C:\Program Files\MATLAB71\work\myGUI.m". The window has a menu bar with "File", "Edit", "Text", "Cell", "Tools", "Debug", "Desktop", "Window", and "Help". Below the menu bar is a toolbar with various icons for file operations, editing, and debugging. The main editing area displays MATLAB code. Line 75 is empty. Line 76 contains a comment: "% --- Executes on button press in pushbutton1." Line 77 contains the function definition: "function pushbutton1_Callback(hObject, eventdata, handles)". Line 78 contains a comment: "% hObject handle to pushbutton1 (see GCBO)". Line 79 contains a comment: "% eventdata reserved - to be defined in a future version of MATLAB". Line 80 contains a comment: "% handles structure with handles and user data (see GUIDATA)". Line 81 is empty. Line 82 is empty. Line 83 contains a comment: "% --- Executes on button press in pushbutton2." Line 84 contains the function definition: "function pushbutton2_Callback(hObject, eventdata, handles)". The status bar at the bottom shows "myGUI / pushbutton1_Callback", "Ln 77", "Col 59", and "OVR".

```
75
76 % --- Executes on button press in pushbutton1.
77 function pushbutton1_Callback(hObject, eventdata, handles)
78 % hObject    handle to pushbutton1 (see GCBO)
79 % eventdata  reserved - to be defined in a future version of MATLAB
80 % handles    structure with handles and user data (see GUIDATA)
81
82
83 % --- Executes on button press in pushbutton2.
84 function pushbutton2_Callback(hObject, eventdata, handles)
```

按钮1“sphere”的回调程序为：

sphere;

axis tight;(axis tight是使坐标系的最大值和最小值和你的数据范围一致)

按钮2“peaks”的回调程序为：

peaks;

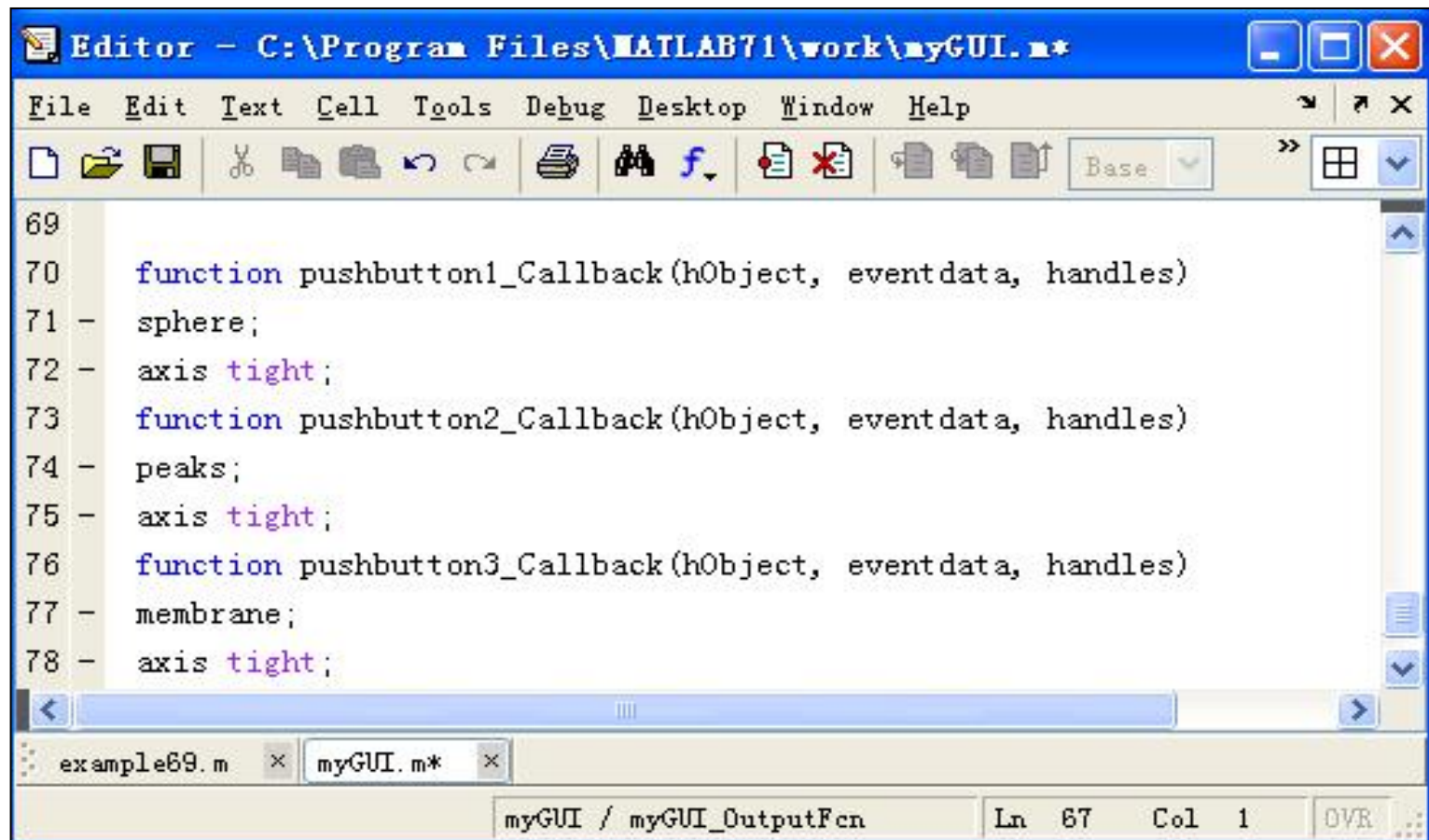
axis tight;

按钮3“membrane”的回调程序为：

membrane;

axis tight;

M文件编辑器中的三个按钮的回调子函数



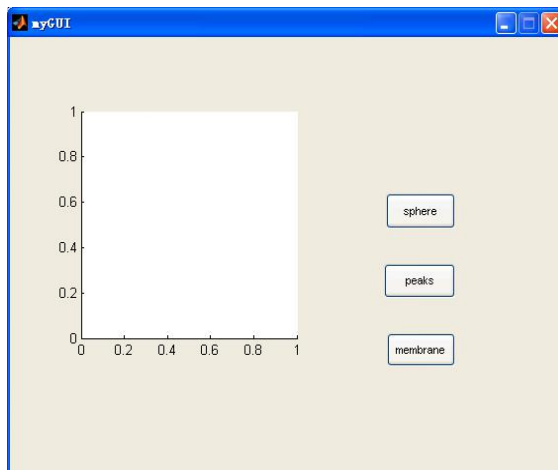
The image shows a screenshot of the MATLAB Editor window. The title bar reads "Editor - C:\Program Files\MATLAB71\work\myGUI.m*". The menu bar includes File, Edit, Text, Cell, Tools, Debug, Desktop, Window, and Help. The toolbar contains various icons for file operations, editing, and debugging. The main text area displays the following code:

```
69  
70     function pushbutton1_Callback(hObject, eventdata, handles)  
71 -     sphere;  
72 -     axis tight;  
73     function pushbutton2_Callback(hObject, eventdata, handles)  
74 -     peaks;  
75 -     axis tight;  
76     function pushbutton3_Callback(hObject, eventdata, handles)  
77 -     membrane;  
78 -     axis tight;
```

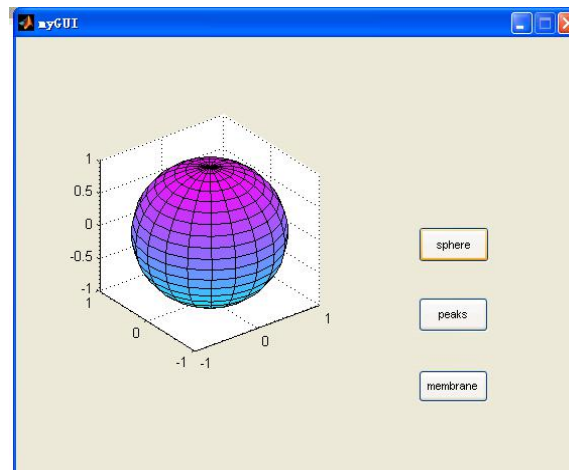
The status bar at the bottom shows "myGUI / myGUI_OutputFcn", "Ln 67 Col 1", and "OVR".

运行结果

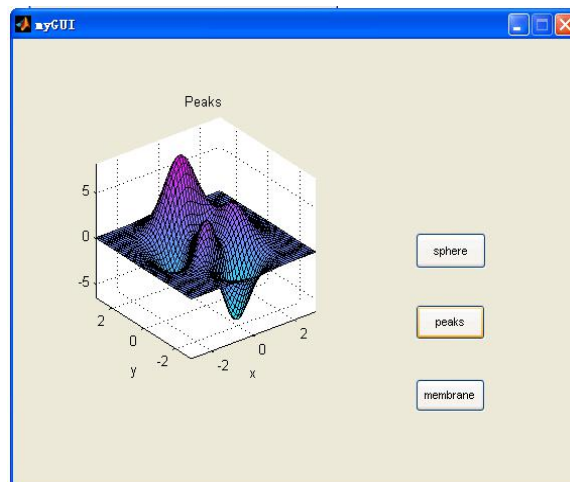
(a)



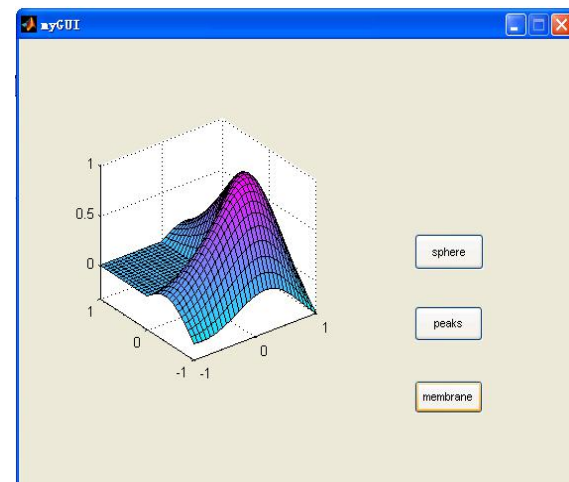
(b)



(c)



(d)



(a) 被激活后的界面； (b) sphere图； (c) peaks图； (d) membrane图