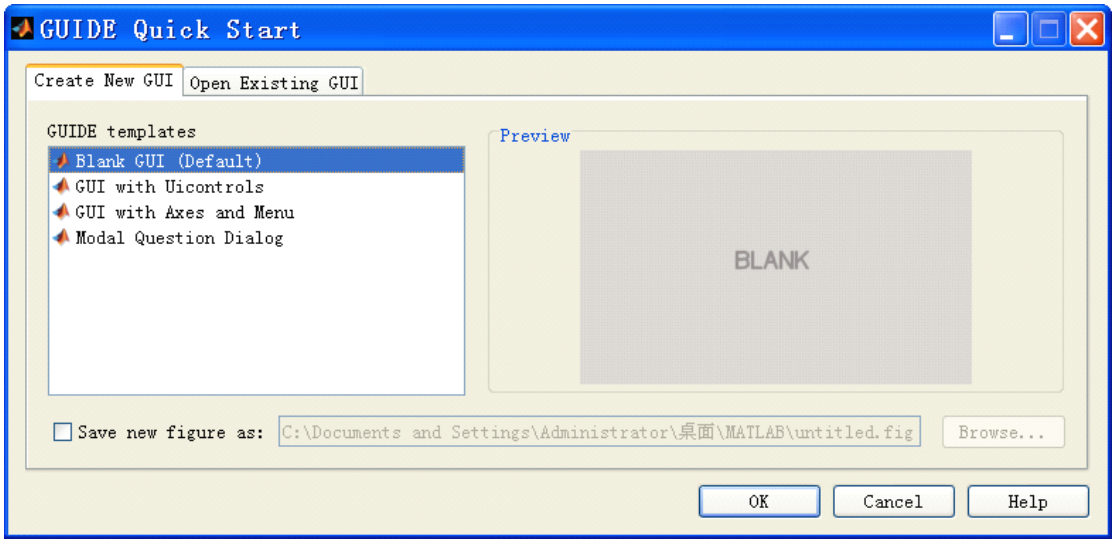
一个实例搞定 MATLAB 界面编程

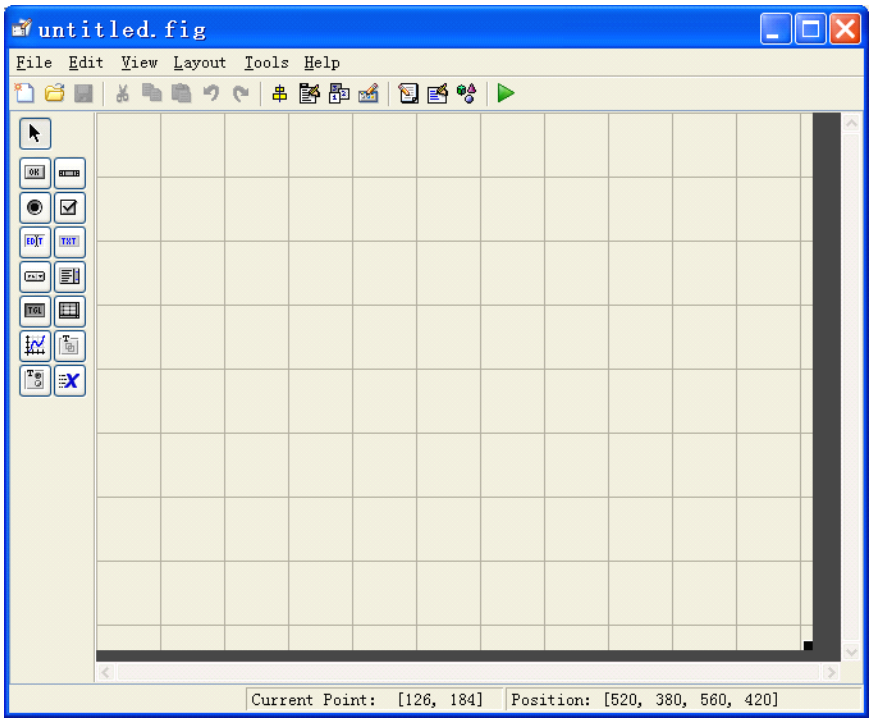
作者：彭军
邮件：pjun9@foxmail.com
博客：http://hi.baidu.com/pengjun

下面请跟我一步一步做一个图像处理的程序，如果您坚持做完这个实例，我想 MATLAB 界面编程对您而言，就没有什么难度了。当然，我这里说的是，您首先要有一定的 MATLAB 编程基础。还有，我的 MATLAB 版本是 2008a。在 2008a 以前的版本中没有工具栏编辑器，如果需要工具栏要手动写程序，这个我就不多讲了。好了，废话少说，跟我来吧！

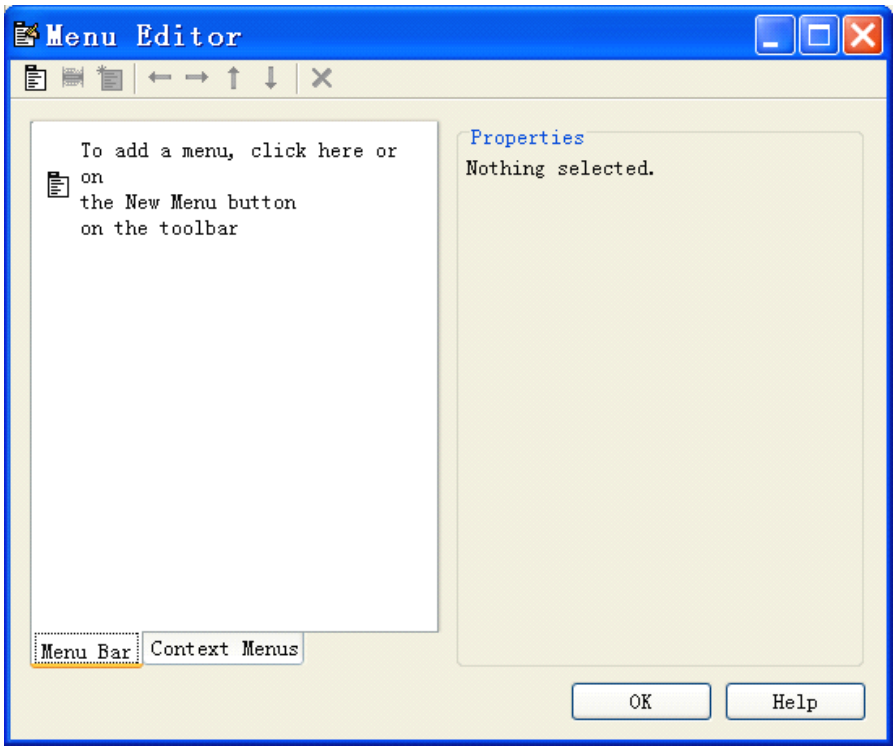
在 MATLAB 的命令窗口(Command Window)中运行 guide 命令，来打开 GUIDE 界面，如下：



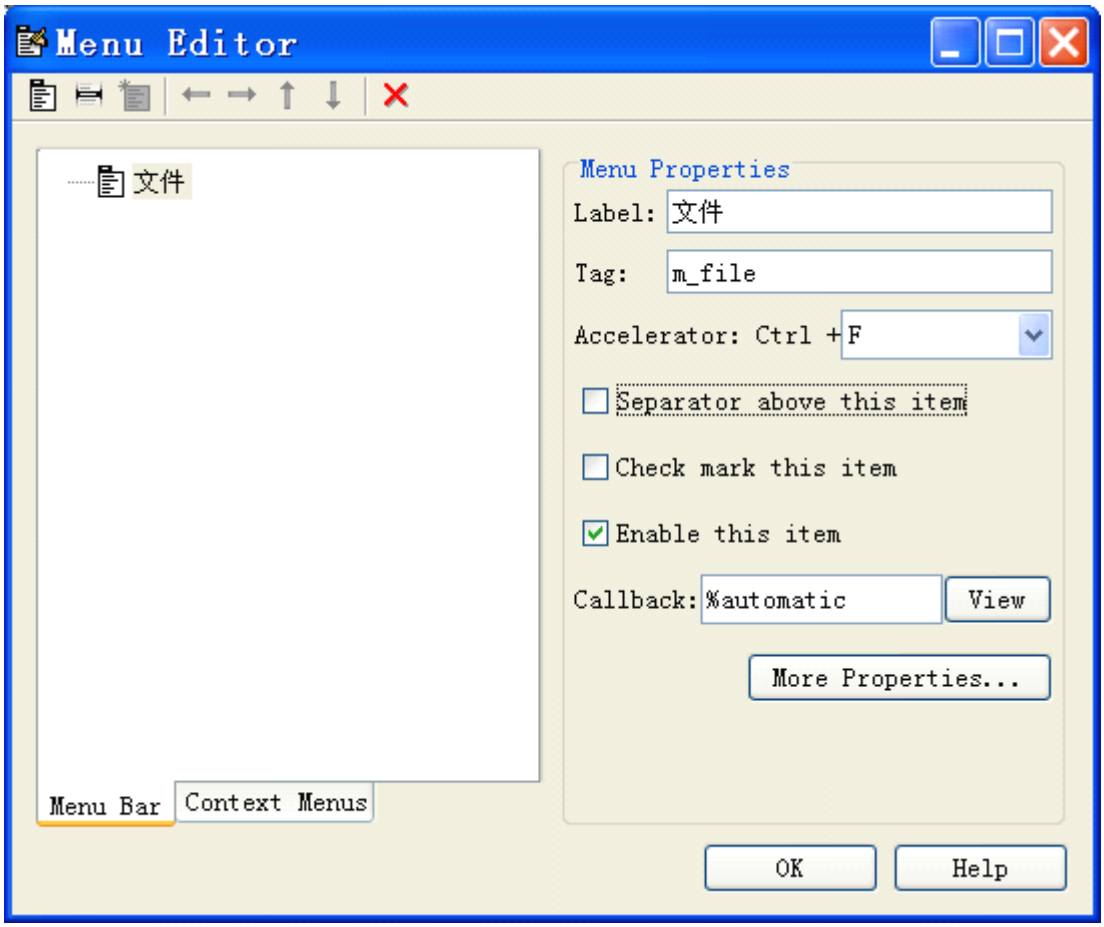
然后，选择空模板(Blang GUI)，点击 OK，即可打开 GUIDE 的设计界面，如下：



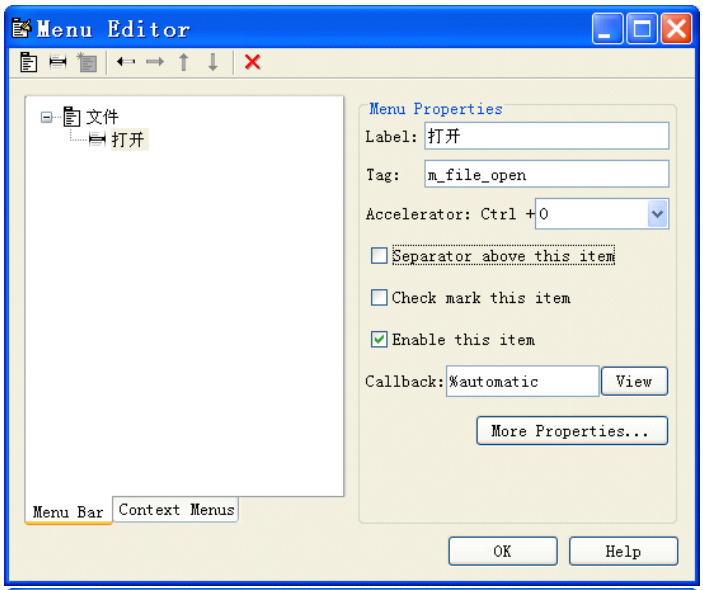
点击工具栏上的菜单编辑器(Menu Editor)，打开菜单编辑器，如下：

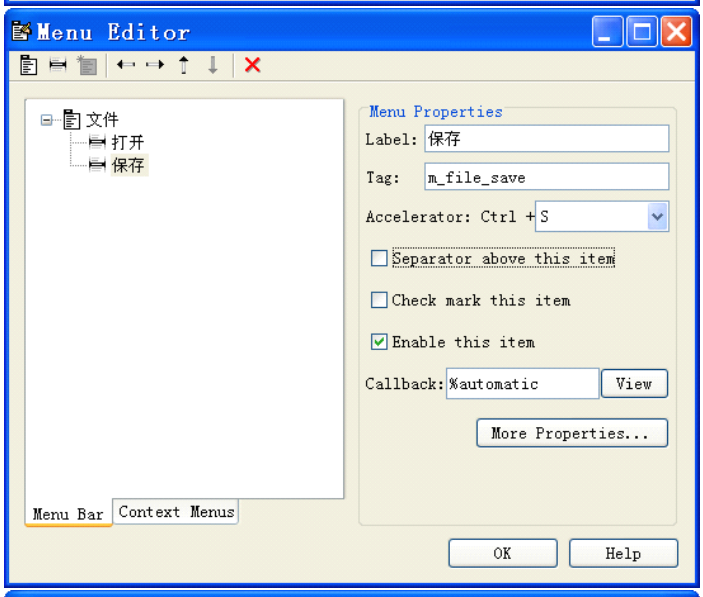


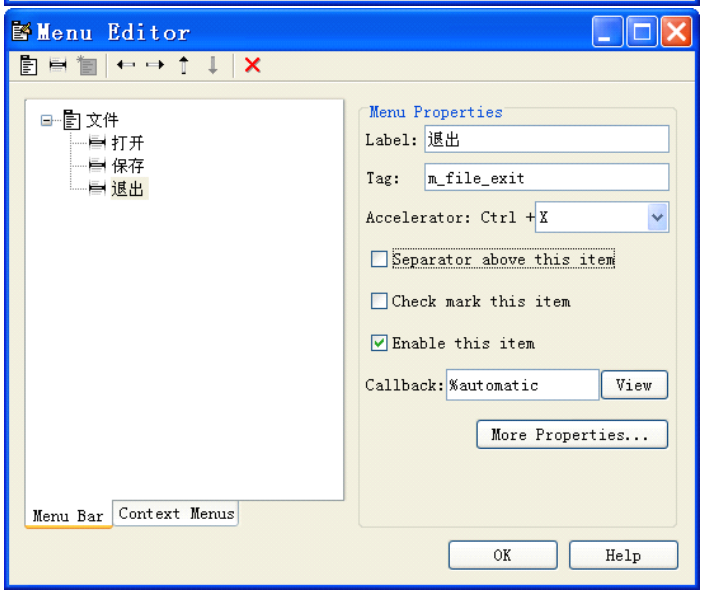
在 Menu Bar 中新建一个菜单项，名字为“文件”，其他设置请看下图：



在“文件”菜单下添加菜单项：“打开”，“保存”，“退出”。见下图：





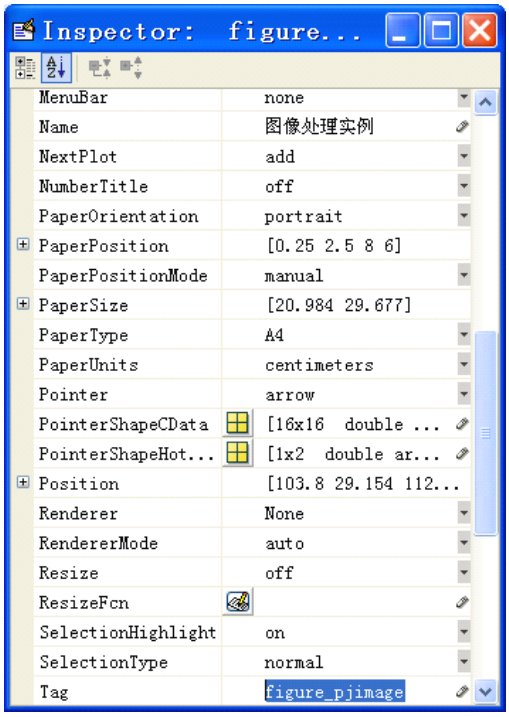


如果需要在菜单项“退出”上面添加一个分割线的话，选中“Separator above this item”就行了。

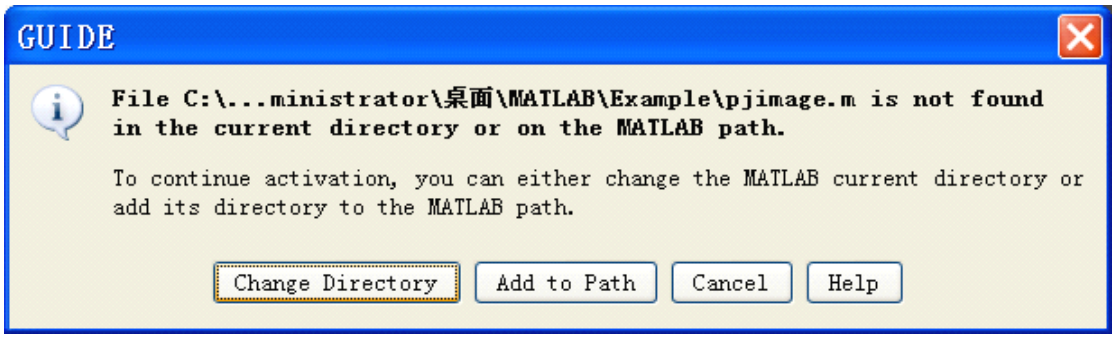
保存我的界面为 pjimage.fig. 保存完毕之后，会自动打开 pjimage.m 文件，而我们所有的程序都是要写在这个 M 文件里面的。在编程中，我们的每一个鼠标动作都对应一个 Callback 函数。那么我们的菜单项也是如此的。

在界面上，单击鼠标右键选择“Property Inspector”，即可打开属性窗口。当我们点击不同的控件时，其对应的属性都会在这里显示，我们可以进行修改。最主要的属性莫过于 Tag 属性和 String 属性。

设置当前 Figure 窗口的 Tag 属性为：figure_pjimage，窗口的标题(Name 属性)为：图像处理实例。如下：



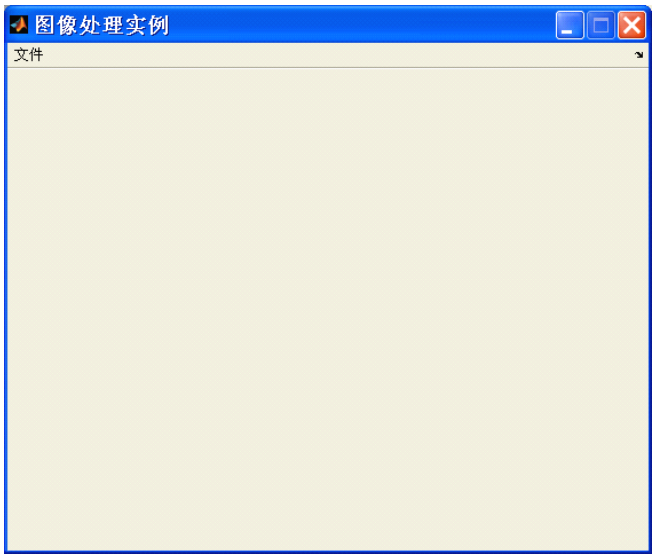
然后，点击工具栏的保存按钮。之后，点击工具栏的运行按钮(Run Figure)。注意，工具栏的图标都会有提示的，像运行按钮的提示就是 Run Figure. 我们会看到如下的界面：



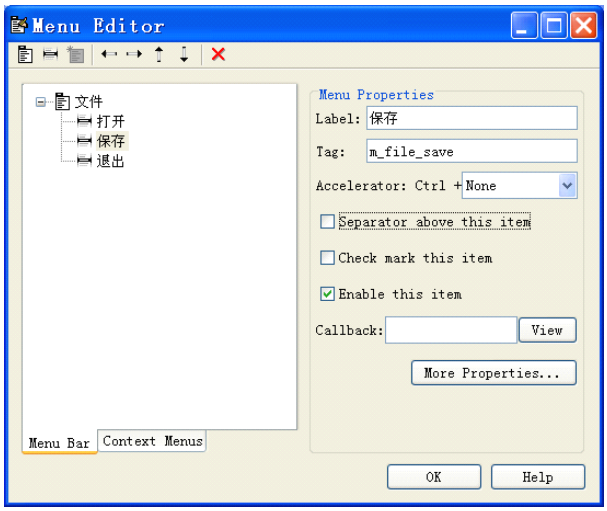
那说明，我们保存的.fig 文件的目录不是当前目录，但是没关系啊，我们只要点击“Change Directory”来改变当前目录。当然，如果你想把当前目录添加到 MATLAB 路径也可以，那就点击“Add to Path”就 OK 了。我在这里推荐点击“Change Directory”，因为没有什么太大必要把其添加到 MATLAB 路径中，一般是工具箱需要添加或者我们的函数或程序写完了，而在 MATLAB 的命令窗口找不到我们的函数的时候，我们可以将函数或程序所在的目录添加到 MATLAB 路径。

总之吧，点那个按钮，要看个人的爱好了。不管点击两个按钮的那一个按钮，都会正确的运行程序的。

我们的程序运行时的样子，是这样的：



文件下面的菜单项和快捷键我们都能看到，但是我们没有写程序，所以就算点也没有什么响应。还有如果不想设置快捷键，可以在 Menu Editor 中设置，只要把其选择为 Ctrl+none 就行了，如下：

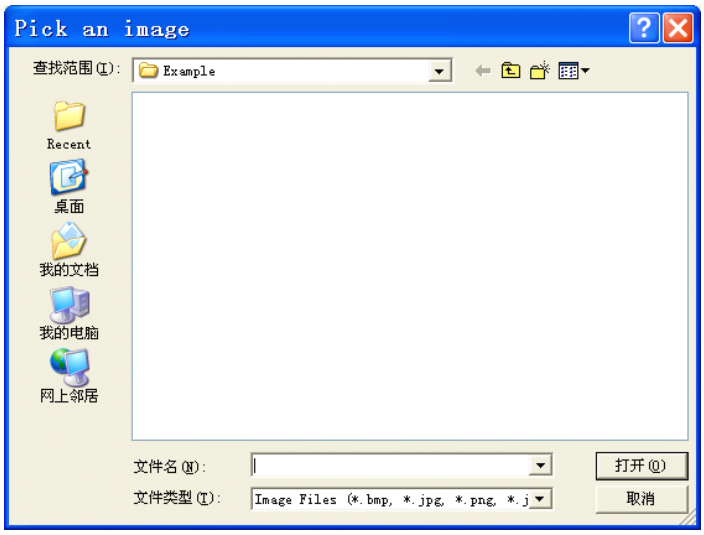


这样的话，保存项就没有了快捷键了。我们可以通过上面的按钮“View”来查看该菜单项的响应函数，也就是 Callback 函数。也可以在 pjimage.m 中看，比如保存的 Tag 属性是 m_file_save,那么它对应的 Callback 函数的名字就是 m_file_save_Callback。依次类推了。

下面我们来写打开菜单项的函数，要打开一个图片，当然要用打开对话框了。在界面编程中，打开对话框的函数是 uigetfile. 关于它的详细的说明用 help uigetfile 命令查看。下面是打开菜单的响应函数：

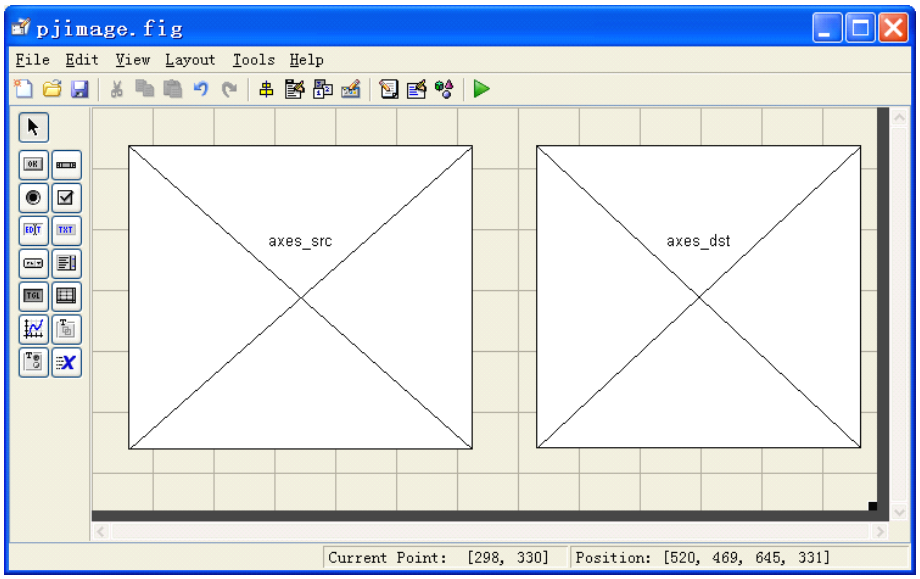
```
function m_file_open_Callback(hObject, eventdata, handles)
 [filename, pathname] = uigetfile( ...
        {'*.bmp;*.jpg;*.png;*.jpeg', 'Image Files (*.bmp, *.jpg, *.png,
 *.jpeg)'; ...
         '*.*',                      'All Files (*.*)'}, ...
         'Pick an image');
```

保存.m 文件，并运行程序。点击“文件”下的“打开"，会打开如下的打开对话框：



选择一个文件之后，程序中的 filename 就是你选择的文件的文件名，pathname 就是该文件所在的目录的路径。比如：filename =5.jpg，pathname =C:\Documents and Settings\Administrator\My Documents\。

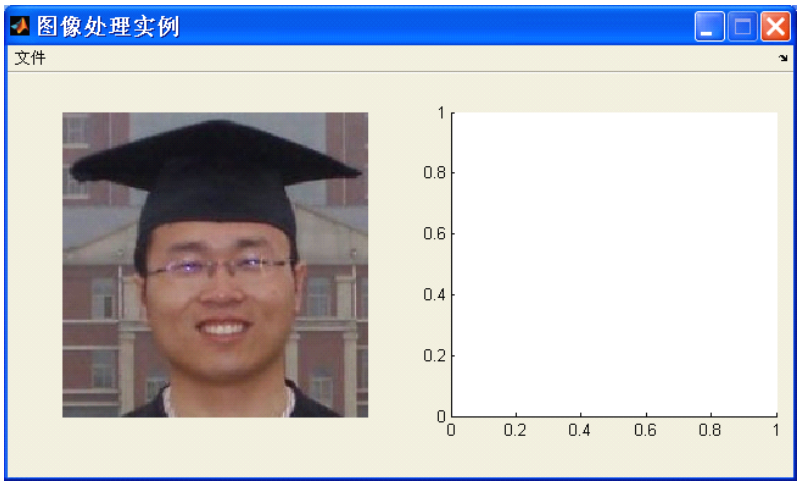
那么获得路径之后，我们要怎么样才能读入和显示一个图片呢？读入图片可以用 imread 函数，而显示可以在一个坐标轴上。那么我们需要在界面上画上一个坐标轴，为了对比，我们画两个坐标轴，一个显示处理前，一个显示处理后的。并且将处理前的坐标轴的 Tag 属性改为 axes_src,处理后的坐标轴的 Tag 属性为 axes_dst。更改之后，保存。如下：



然后在 m_file_open_Callback 程序原来的基础上，再添加如下的程序：

```
axes(handles.axes_src);%用axes命令设定当前操作的坐标轴是axes_src
fpath=[pathname filename];%将文件名和目录名组合成一个完整的路径
imshow(imread(fpath));%用imread读入图片，并用imshow在axes_src上显示
```

运行程序，通过“打开”菜单项，打开一个图片。效果如下：



那么如何来保存一副图片？用imwrite命令。但imwrite命令的第一个参数就是你读入的图片数据，也就是imread的返回值。这样的话，我们就要将m_file_open_Callback中的程序做一点小小的改动。将最后一句(imshow(imread(fpath)))，更改为两句，如下：

```
img_src=imread(fpath);imshow(img_src);
```

不仅如此，我们的保存菜单的Callback函数，如何去获得打开菜单的Callback函数下的img_src变量呢？这里就要将img_src来作为一个共享的数据。许多界面编程的朋友，喜欢用global声明。我个人不喜欢这样用，因为有更好的方法。那就是用setappdata和getappdata两个函数。我们可以为界面上面的任何一个具有Tag属性的空间添加应用程序数据。当然我比较喜欢将这些共享的应用程序数据统一添加到Figure窗口上，因为这样容易记，如果一个控件一个，感觉不容易记。你在.m文件中会发现除了各个菜单项的Callback函数以外，还有两个函数：pjimage_OpeningFcn和pjimage_OutputFcn.而pjimage_OpeningFcn就相当于界面的初始化函数，而pjimage_OutputFcn则是界面的输出函数，也就是当你不运行fig，而调用.m文件时的返回值。

所以，我们要在pjimage_OpeningFcn中添加如下的程序，来共享这个img_src矩阵。代码如下：

```
setappdata(handles.figure_pjimage,' img_src' ,0);
```

然后，在m_file_open_Callback函数的最后写上如下程序：

```
setappdata(handles.figure_pjimage,' img_src' ,img_src);
```

那么，我们在m_file_save_Callback函数中就可以像这样的来提取img_src，如下：

```
img_src=getappdata(handles.figure_pjimage,' img_src' );
```

那么保存的时候，自然会用到保存对话框了。要用保存对话框，就要用到uiputfile函数了，具体的请用help uiputfile查看。

那么，保存菜单项下的程序(m_file_save_Callback)，可以这样写：

```
[filename, pathname] = uiputfile({'*.bmp','BMP files';'*.jpg;','JPG files'}, 'Pick an Image');
if isequal(filename,0) || isequal(pathname,0)
    return;%如果点了“取消”
else
    fpath=fullfile(pathname, filename);%获得全路径的另一种方法
```

```
end
img_src=getappdata(handles.figure_pjimage,'img_src');%取得打开图片的数据
imwrite(img_src,fpath);%保存图片
```

下面是退出菜单项的程序的。要退出界面，只要用close函数就行了，但是通常都会有提示的。比如你如果进行了处理图片，而又没有保存处理后的图片，那么在关闭的时候就应该给出提示，询问是否进行保存。不过，在这里，我们先不做这个工作，等后面有需要的时候再写吧。因此，这里的退出菜单项的程序就是一句，如下：

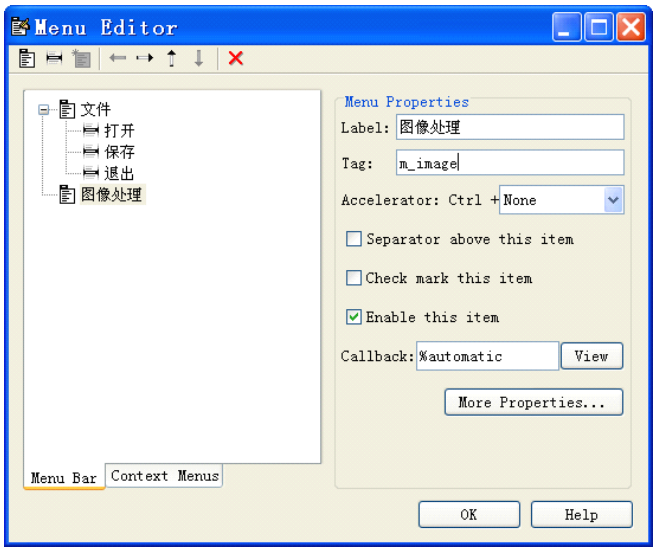
```
close(handles.figure_pjimage);
```

其实，用delete函数也是可以的，就是：delete(handles.figure_pjimage);看你的心情了。

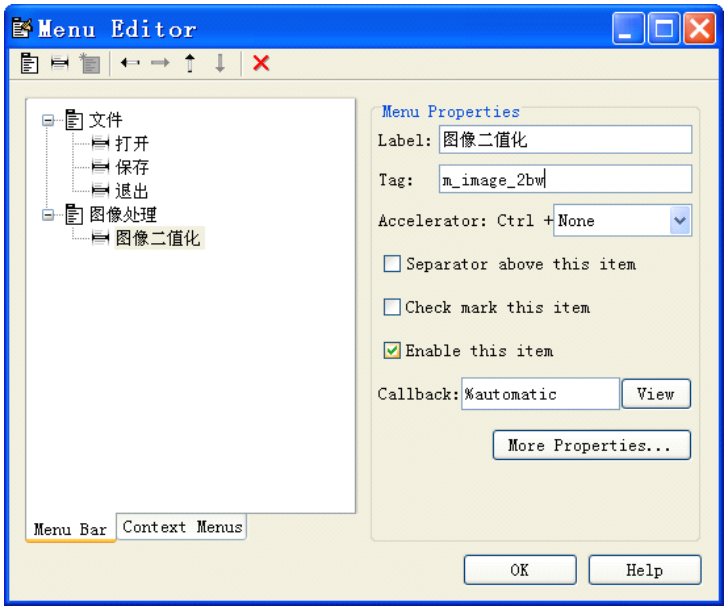
但是运行程序的时候，你会发现，当你打开图片的时候，如果点“取消”按钮，那么在MATLAB的命令窗口会弹出错误，那是因为我们没有处理取消的情况。下面我们来处理下这个问题，只要把m_file_open_Callback下面的程序更改为如下程序即可：

```
[filename, pathname] = uigetfile( ...
    {'*.bmp;*.jpg;*.png;*.jpeg', 'Image Files (*.bmp, *.jpg, *.png, *.jpeg)'; ...
    '*.*',                       'All Files (*.*)'}, ...
    'Pick an image');
if isequal(filename,0) || isequal(pathname,0),
    return;
end
axes(handles.axes_src);
fpath=[pathname filename];
img_src=imread(fpath);
imshow(img_src);
setappdata(handles.figure_pjimage,'img_src',img_src);
```

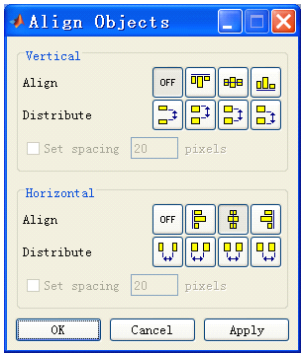
下面我们来做一个图像二值化的一个图像处理。用上面的方法添加一个“图像处理”菜单，如下：



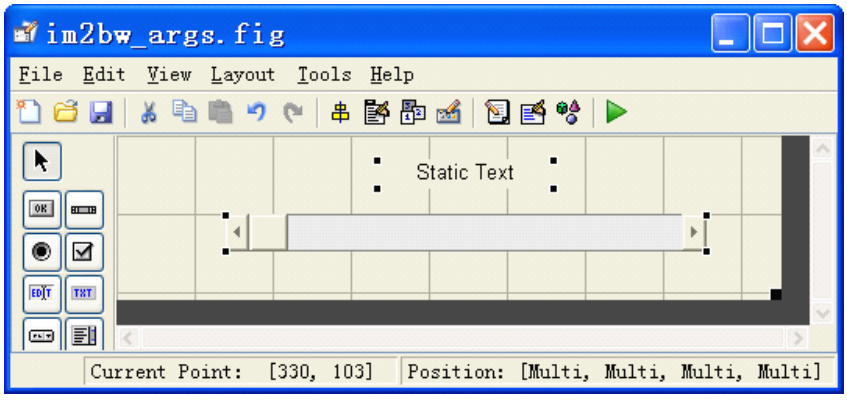
在其下面添加一个“图像二值化”的菜单项，如下：



然后，点击“OK”关闭菜单编辑器，并保存整个界面。如果我们的.m文件中没有对应的Callback时，我们可以点击上图中的“View”按钮来生成一个Callback函数。图像二值化，有一个阈值的设置，那么我们可以新建一个界面，在这个界面上放一个滑动条来设置图像二值化的阈值。同时，有一个文本，显示当前滑动条的值。那么我们新建一个空白界面，在它上面画一个Static Text和Slider控件，然后用工具栏的对齐工具(Align Objects)，来对其这两个空间。如下：



然后，将这个界面保存为im2bw_args.fig。整个设计如下：

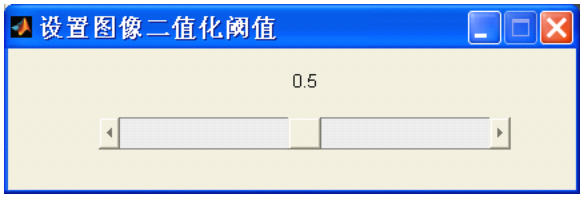


你可以设置Static Text的FontSize属性为10，这样字体会更大一点。设置Static Text的Tag属性为txt_display，设置滚动条的Tag属性为slider_val。为了能够在滚动条滚动时，Static Text显示滚动条的值，需要在滚动条的Callback中写下

如下程序，你可以在滚动条上点击右键，选择“View Callbacks”下的“Callback”直接进入滚动条的Callback函数(slider_val_Callback)。

```
val=get(hObject,'Value');
set(handles.txt_display,'String',num2str(val));
```

保存，运行程序，就可以滑动滚动条，而Static Text就会显示相应的值。在figure上双击打开figure(有方块的底层窗口)的属性窗口，将其Tag属性设置为“figure_im2bw”，将其Name属性设置为“设置图像二值化阈值”。然后，保存界面。运行时，如下：



那么，我们想的是，当滑动条滑动时，将二值化的图像显示在pjimage.fig中的axes_dst坐标轴上的。那么怎么办呢？首先，要做的是，当点击pjimage.fig菜单“图像处理”下的“图像二值化”的时候，会打开im2bw_args.fig。这个时候就是我们要调用im2bw_args.m的时候了。当我们调用它的时候，会返回一个句柄，而这个句柄就是指向打开的im2bw_args.fig的。关于更详细的，你可以参看im2bw_args.m文件的最前面的注释，其中有这样写：

```
% H = IM2BW_ARGS returns the handle to a new IM2BW_ARGS or the handle
to
% the existing singleton*.
```

那就说明，我们可以如上的方式打开im2bw_args.fig。所以在“图像二值化”的Callback函数(m_image_2bw_Callback)下，写上如下的程序：

```
h=im2bw_args;
```

然后，保存pjimage.fig.还有就是，最好将im2bw_args.fig和pjimage.fig保存在一个目录下面。然后，运行pjimage.fig，可以看到，当点击“图像二值化”的时候会打开im2bw_args.fig,同时滑动条滑动时也会显示响应的值。

下面来说说如何在滑动条滑动时，将滑动后的二值化图像显示到pjimage的axes_dst坐标轴中。

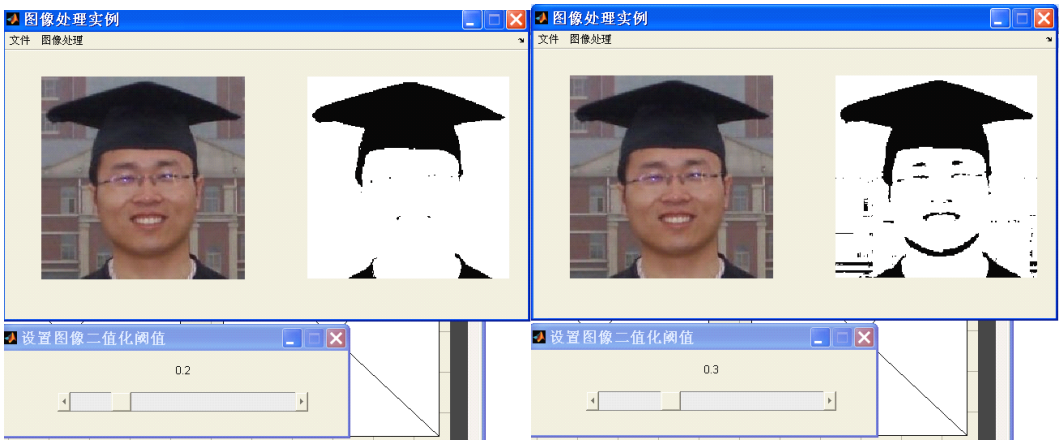
首先，我们要获得pjimage的figure的句柄，这个可以通过findobj函数来完成，之后将返回值用guihandles来转换成一个句柄。之后，就可以用这个转化后的句柄来引用pjimage.fig中的任何一个控件了。所以，我们在im2bw_args.fig下的滑动条的Callback函数中添加如下函数：

```
h_pjimage=getappdata(handles.figure_im2bw,'h_pjimage');
axes(h_pjimage.axes_dst);
img_src=getappdata(h_pjimage.figure_pjimage,'img_src');
bw=im2bw(img_src,val);
imshow(bw);
```

然后，在im2bw_args_OpeningFcn中添加：

```
h_pjimage=findobj('Tag','figure_pjimage');
h_pjimage=guihandles(h_pjimage);
setappdata(handles.figure_im2bw,'h_pjimage',h_pjimage);
```

然后，保存，运行。效果如下：



但是，如果在我们没有打开图片的情况下，要是点击了“图像二值化”会出现什么问题呢？可以看到显示的图像是全黑的，完全没有意义。所以，我们可以在没有点击“打开”菜单项的时候，使“图像处理”菜单不可用。

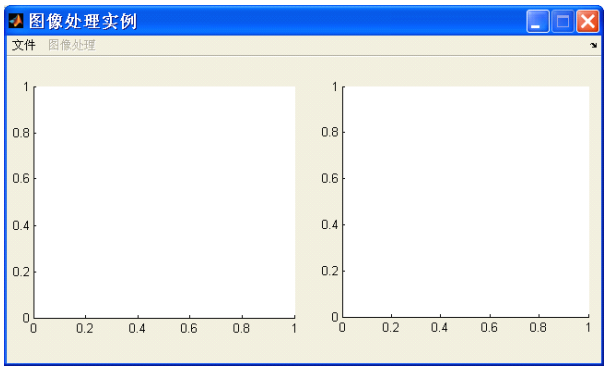
那么在pjimage.m的OpeningFcn中，添加如下程序：

```
set(handles.m_image,'Enable','off');
```

在“打开”菜单项的Callback函数的最后，添加如下程序：

```
set(handles.m_image,'Enable','on');
```

这样的话，只要你不点“打开”，就不能用“图像处理”菜单中的命令，效果如下：



点击“打开”之后，就能使用了。

下面，我们来说说前面的问题，就是询问是否保存图片的问题。首先，我们要设置两个标志：一个是图片是否被处理过了，二是图片是否被保存了。那么我们在pjimage_OpeningFcn中，添加如下的两个应用程序数据。

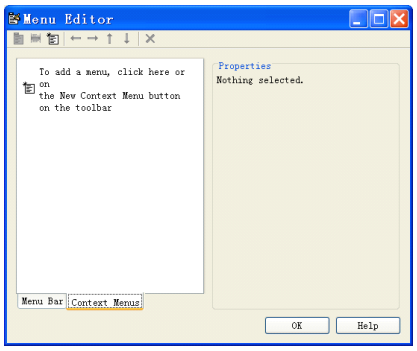
```
setappdata(handles.figure_pjimage,'bSave',false);
setappdata(handles.figure_pjimage,'bChanged',false);
```

然后在“图像二值化”菜单项的Callback函数中，改变bChanged的值为true，即添加如下程序：

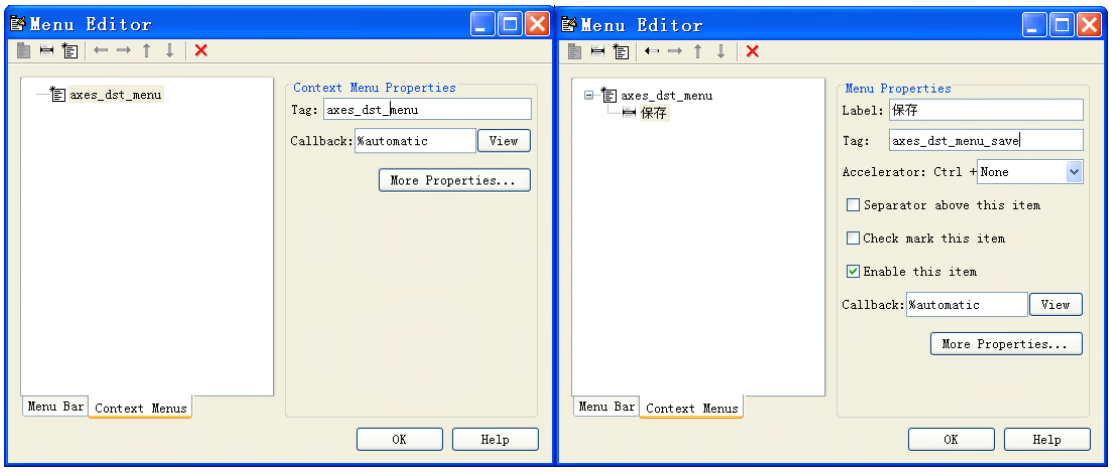
```
setappdata(handles.figure_pjimage,'bChanged',true);
```

由于我们要保存的是坐标轴axes_dst中的图像，而我们“文件”下的“保存”，实质上保存的是坐标轴axes_src中的图像，那怎么办呢？只好再添加一个“保存”菜单项了。这次，我们在坐标轴axes_dst中添加右键菜单。

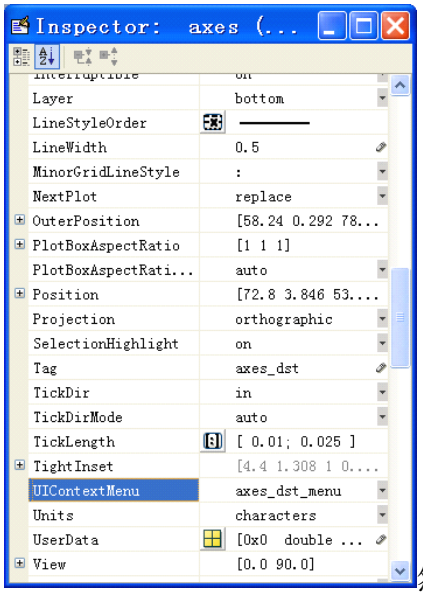
打开工具栏的菜单编辑器，选择Context Menu(上下文菜单)，如下：



然后，新建一个Context Menu,其Tag属性为：axes_dst_menu,如下：



然后为其添加菜单项：“保存”，其Tag属性为axes_dst_menu_save.如上图。然后，在坐标轴axes_dst上右键，选择“Property Inspector”。将该坐标轴的UIContextMenu属性更改为axes_dst_menu. 如下图：



然后，保存，运行。在axes_dst上点右键就能看到“保存”菜单了。下面来写其函数。

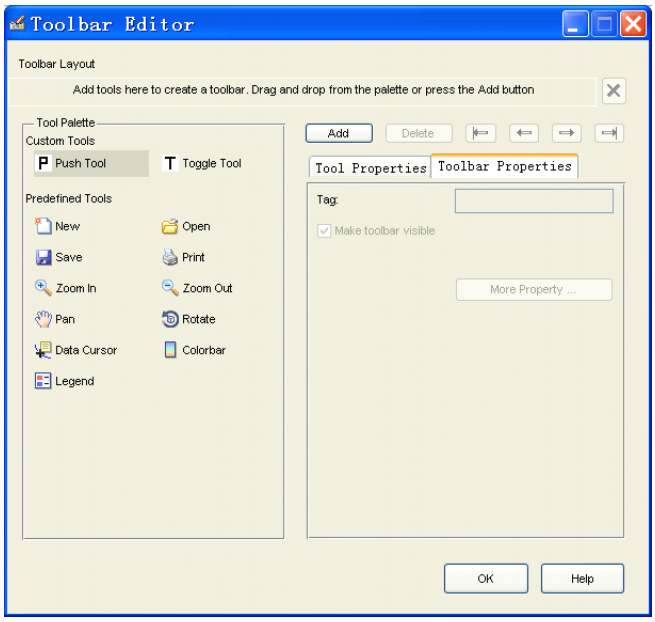
```
[filename, pathname] = uiputfile({'*.bmp','BMP files';'*.jpg;','JPG files'}, 'Pick an Image');
if isequal(filename,0) || isequal(pathname,0)
```

```
    return;
else
    fpath=fullfile(pathname, filename);
end
img_dst=getimage(handles.axes_dst);
imwrite(img_dst,fpath);
setappdata(handles.figure_pjimage,'bSave',true);
```
但是你会发现，没有读入图片之前，在axes_dst点右键是有菜单的，一旦二值化之后，再次点右键就没有菜单了。

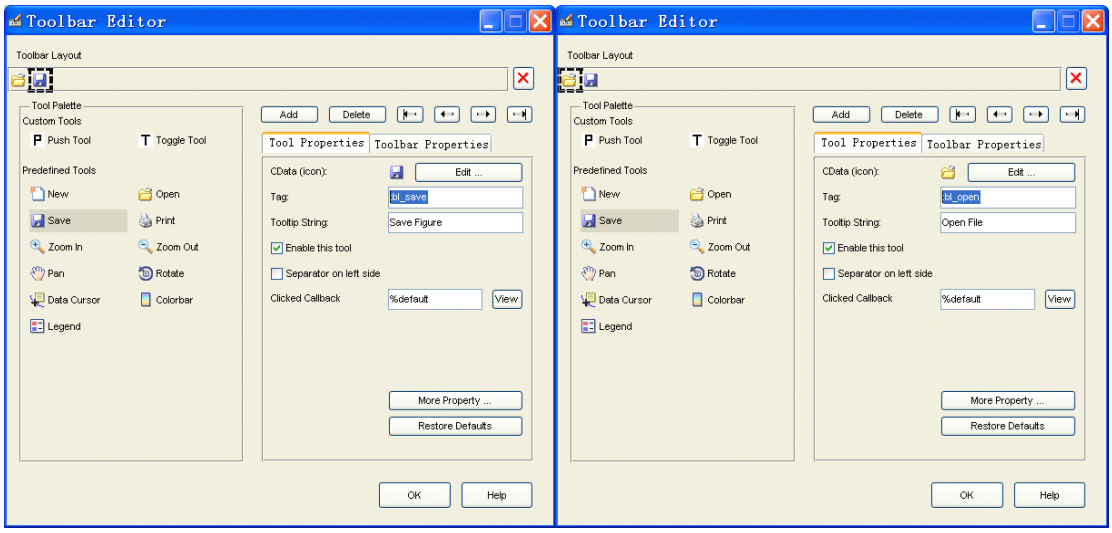
但是，当我们把右键菜单axes_dst_menu,添加到figure窗口(在没有控件的地方，双击，即可打开figure的属性窗口)的UIContextMenu的时候，就不会出现上面的问题，而且一切运行正常。因为，当你添加到axes_dst之后，一旦坐标轴的内容改变，就会将右键菜单附加到父对象上。因此，如果一定需要在坐标轴上显示右键菜单，就要通过程序创建了。如何创建，咱们先不说，先说说把坐标轴axes_dst保存完毕，退出程序的时候的处理。

将原来的m_file_exit_Callback更改为如下程序：

```
bChanged=getappdata(handles.figure_pjimage,'bChanged');%获得是否更改
bSave=getappdata(handles.figure_pjimage,'bSave');%获得是否保存
if bChanged==true && bSave==false,%更改了，而没保存时
    btnName=questdlg('您已经更改了图片，但没有保存。要保存吗?','提示','保存','不保存','保存');%用提问对话框
    switch btnName,
        case '保存',    %执行axes_dst_menu_save_Callback的功能
feval(@axes_dst_menu_save_Callback,handles.axes_dst_menu_save,eventdata,handles);
        case '不保存',%什么也不做
    end
end
h=findobj('Tag','figure_im2bw');%查找是否打开设置图像二值化参数窗口
if ~isempty(h),%找到的话，则关闭
    close(h);
end
close(findobj('Tag','figure_pjimage'));%关闭主窗口
```
下面来为程序添加一个工具栏，单击工具栏上那个的Toolbar Editor，打开如下：



选择“Predefined Tools”下的Open，点击“Add”。再次选择“Save”，点击“Add”。并将Open按钮的Tag属性更改为tbl_open,Save按钮的Tag属性更改为tbl_save,如下：



点“View”，来找到Open按钮的Callback，在它的下面来调用菜单中的打开菜单项的Callback，需要在Open按钮的Callback下写下如下程序：

```
feval(@m_file_open_Callback,handles.m_file_open,eventdata,handles);
```

用同样的方法，找到Save按钮的Callback，并在它的下面写上保存程序，但是，我们要判断一下是不是第一次保存，如果是，则用保存对话框；如果不是，我们直接保存在第一次保存的路径中就可以了。那么，我们还是需要设置几个应用程序数据的,第一个就是记录是否是第一次保存,第二个是记录第一次保存的路径。这样的话，我们在pjimage_OpeningFcn中添加如下的代码：

```
setappdata(handles.figure_pjimage,'fstSave',true);
setappdata(handles.figure_pjimage,'fstPath',0);
```

然后，在Save按钮的Callback下，写下如下的程序：

```
fstSave=getappdata(handles.figure_pjimage,'fstSave');
if(fstSave==true)
```

```
    [filename, pathname] = uiputfile({'*.bmp','BMP
files';'*.jpg;','JPG files'}, 'Pick an Image');
    if isequal(filename,0) || isequal(pathname,0)
        return;
    else
        fpath=fullfile(pathname, filename);
    end
    img_dst=getimage(handles.axes_dst);
    imwrite(img_dst,fpath);
    setappdata(handles.figure_pjimage,'fstPath',fpath);
    setappdata(handles.figure_pjimage,'bSave',true);
    setappdata(handles.figure_pjimage,'fstSave',false);
else
    img_dst=getimage(handles.axes_dst);
    fpath=getappdata(handles.figure_pjimage,'fstPath');
    imwrite(img_dst,fpath);
end
```

并且，我们还需要在没有打开图片之前的“文件”下的“保存”和工具栏的“Save”按钮都不可用，只有点击“文件”下的“打开”或工具栏下的“打开”的时候，它们才可用。那么需要在pjimage_OpeningFcn中添加如下代码：

```
set(handles.tbl_save,'Enable','off');
set(handles.m_file_save,'Enable','off');
```

并且在m_file_open_Callback下，添加如下代码：

```
set(handles.tbl_save,'Enable','on');
set(handles.m_file_save,'Enable','on');
```

这样一个小程序，算是完成了。