GUIDE – MATLAB

CONTENIDO

1. INTRODUCCIÓN

GUIDE es un entorno de programación visual disponible en MATLAB para realizar y ejecutar programas que necesiten ingreso continuo de datos.

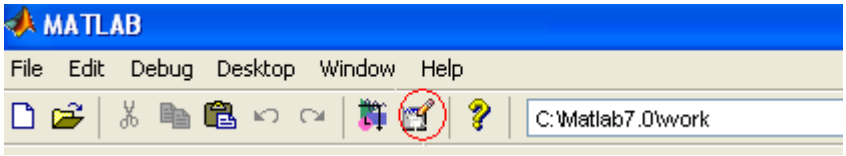
INICIO

Para iniciar nuestro proyecto, lo podemos hacer de dos maneras:

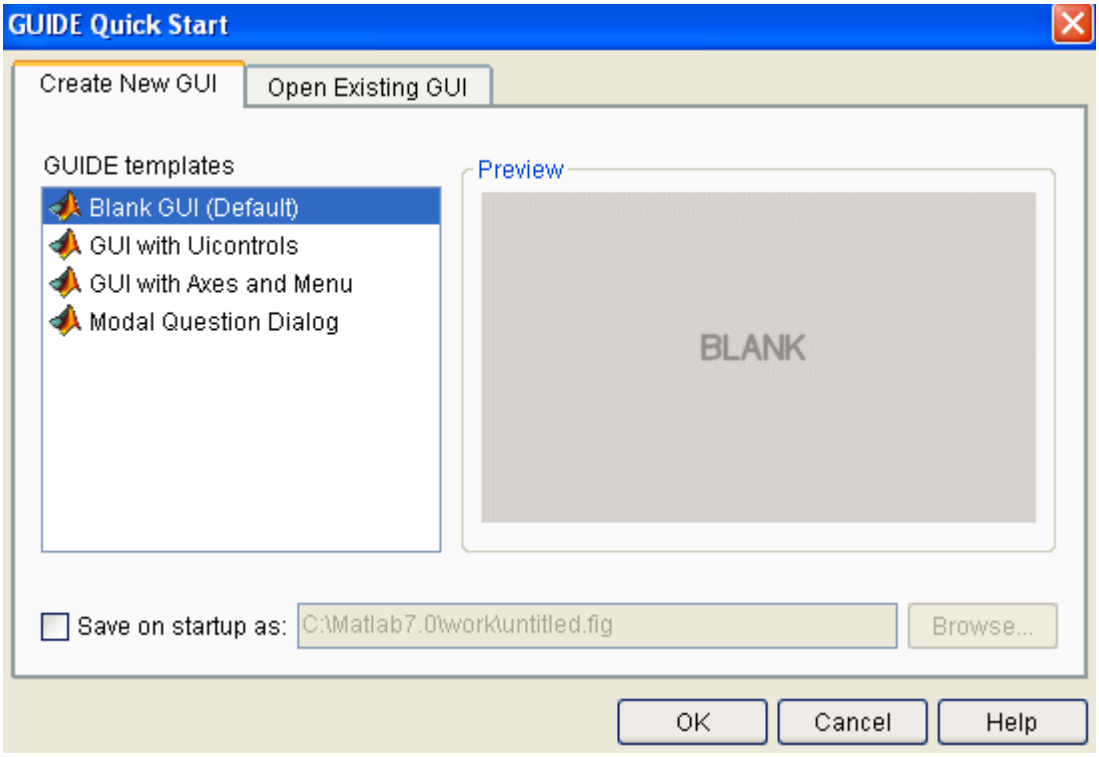
a) Ejecutando la siguiente instrucción en la ventana de comandos:

>> guide

b) Haciendo un click en el ícono que muestra la figura:



Se presenta el siguiente cuadro de diálogo:



Se presentan las siguientes opciones:

a) Blank GUI (Default)

La opción de interfaz gráfica de usuario en blanco (viene predeterminada), nos presenta un formulario nuevo, en el cual podemos diseñar nuestro programa.

b) GUI with Uicontrols

Esta opción presenta un ejemplo en el cual se calcula la masa, dada la densidad y el volumen, en alguno de los dos sistemas de unidades. Podemos ejecutar este ejemplo y obtener resultados.

c) GUI with Axes and Menu

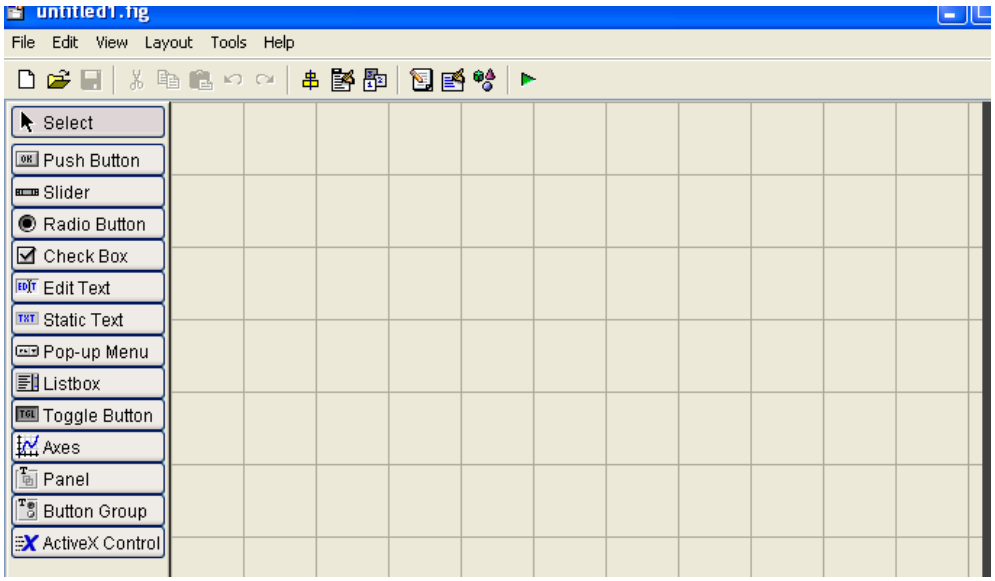
Esta opción es otro ejemplo el cual contiene el menú File con las opciones Open, Print y Close. En el formulario tiene un *Popup menu*, un *push button* y un objeto *Axes*, podemos ejecutar el programa eligiendo alguna de las seis opciones que se encuentran en el menú despegable y haciendo click en el botón de comando.

d) Modal Question Dialog

Con esta opción se muestra en la pantalla un cuadro de diálogo común, el cual consta de una pequeña imagen, una etiqueta y dos botones *Yes* y *No*, dependiendo del botón que se presione, el GUI retorna el texto seleccionado (la cadena de caracteres 'Yes' o 'No').

Para obtener la etiqueta de cada elemento de la paleta de componentes ejecutamos: *File>>Preferentes* y seleccionamos *Show names in component palette.*

Tenemos la siguiente presentación:

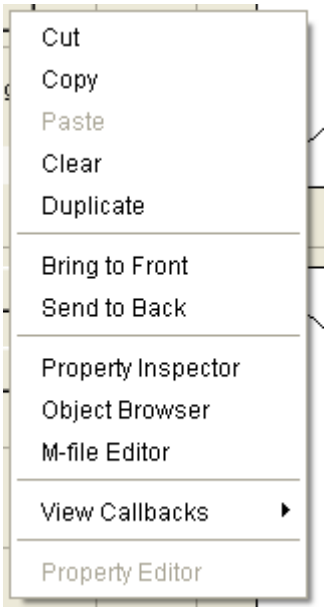


2. COMPONENTES

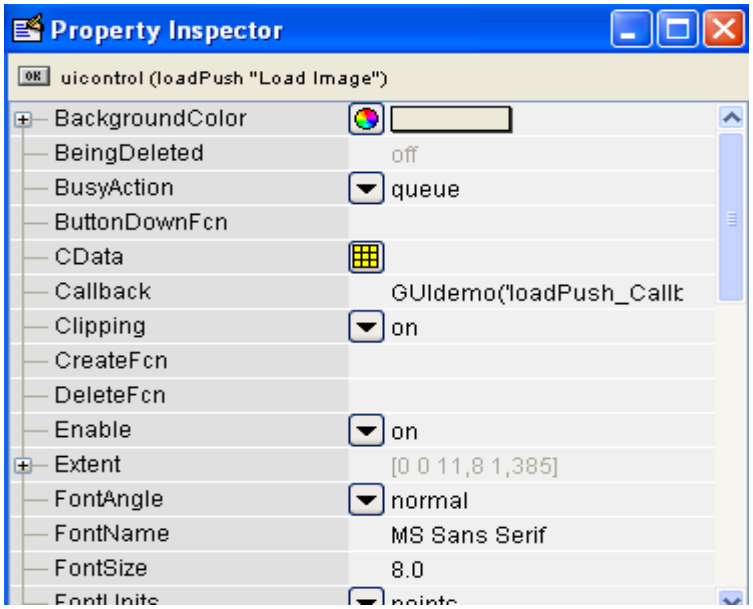
CONTROL	DESCRIPCIÓN
Push Button	Genera una acción
Slider	Representa un rango de valores
Radio Button	Representa una opción
Check Box	Indica el estado de una opción
Edit Text	Para editar texto
Static text	Muestra un string de texto
Pop-up Menu	Provee una lista de opciones
Listbox	Lista deslizable
Toggle Button	Genera una acción on, off
Axes	Para graficar
Panel	Visualiza grupo de controles
Button Grup	Es un panel exclusivo para radio buttons y toggle buttons
ActiveX Control	Despliega controles ActiveX en Gui

PROPIEDADES DE LOS COMPONENTES

Cada uno de los elementos de GUI, tiene un conjunto de opciones que acceder con click derecho. Aparece el siguiente submenú:



La opción *Property Inspector* nos permite personalizar cada elemento.



Al hacer click derecho en el elemento ubicado en el área de diseño, una de las opciones más importantes es *View Callbacks*, la cual, al ejecutarla, abre el archivo *.m. a*sociado a nuestro diseño y nos posiciona en la parte del programa que corresponde a la subrutina que se ejecutará cuando se realice una determinada acción sobre el elemento que estamos editando.

3. FUNCIONAMIENTO DE UNA APLICACIÓN GUI

Una aplicación GUIDE consta de dos archivos: *.m* y *.fig*. El archivo *.m* es el que contiene el código con las correspondencias de los botones de control de la interfaz y el archivo *.fig* contiene los elementos gráficos.

Cada vez que se adicione un nuevo elemento en la interfaz gráfica, se genera automáticamente código en el archivo *.m*. Para ejecutar una Interfaz Gráfica, si la hemos etiquetado con el nombre *curso.fig*, simplemente ejecutamos en la ventana de comandos >> *curso*. O haciendo click derecho en el m-file y seleccionando la opción *RUN*.

SENTENCIAS GET Y SET

La asignación u obtención de valores de los componentes se realiza mediante las sentencias *get* y *set*. Por ejemplo:

```
celsius1=eval(get(handles.celsius,'string'));
%Para convertir celsius a kelvin
kelvin1=celsius1 + 273.15;
```

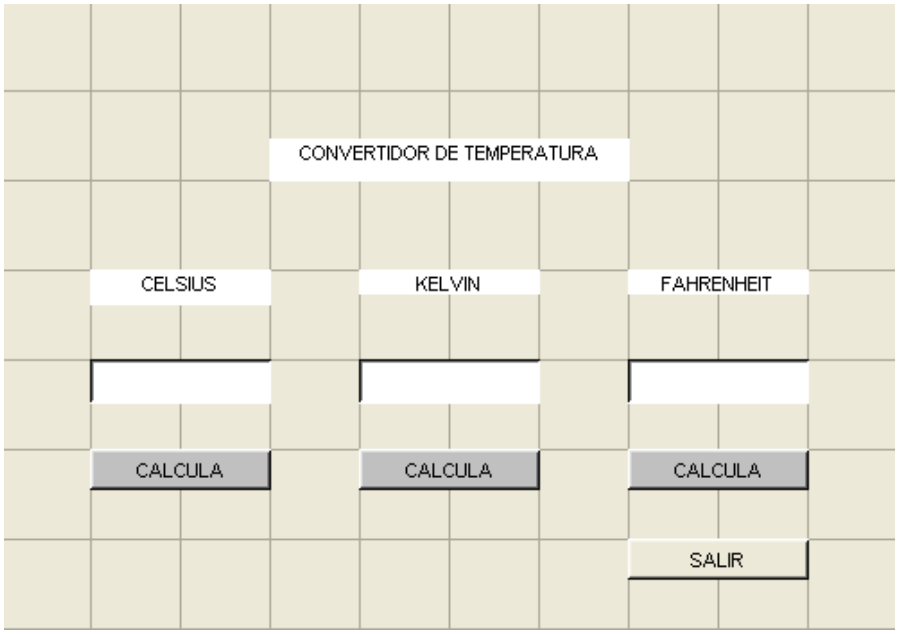
Notar que siempre se obtienen los datos a través de los identificadores *handles*.

Para colocar el valor de la variable *kelvin1* al *statictext,(Tag kelvin)* escribimos:

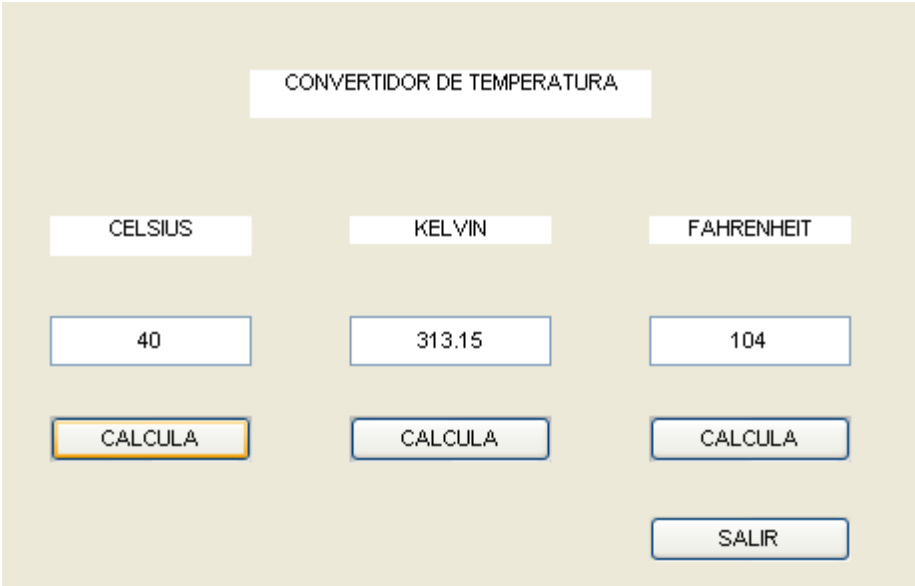
```
set(handles.kelvin,'string',kelvin1);
```

4. EJEMPLOS

Ejemplo: ConvTemperatura.fig



Al correrse el programa ConvTemp, escibimos en la casilla de celsius 40 y calculamos. Se obtiene la siguiente figura:

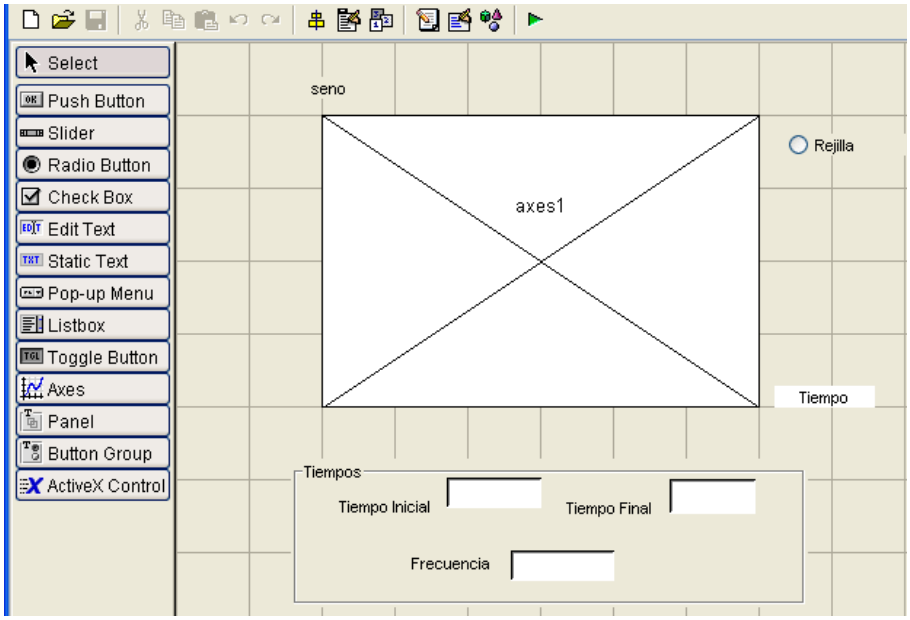


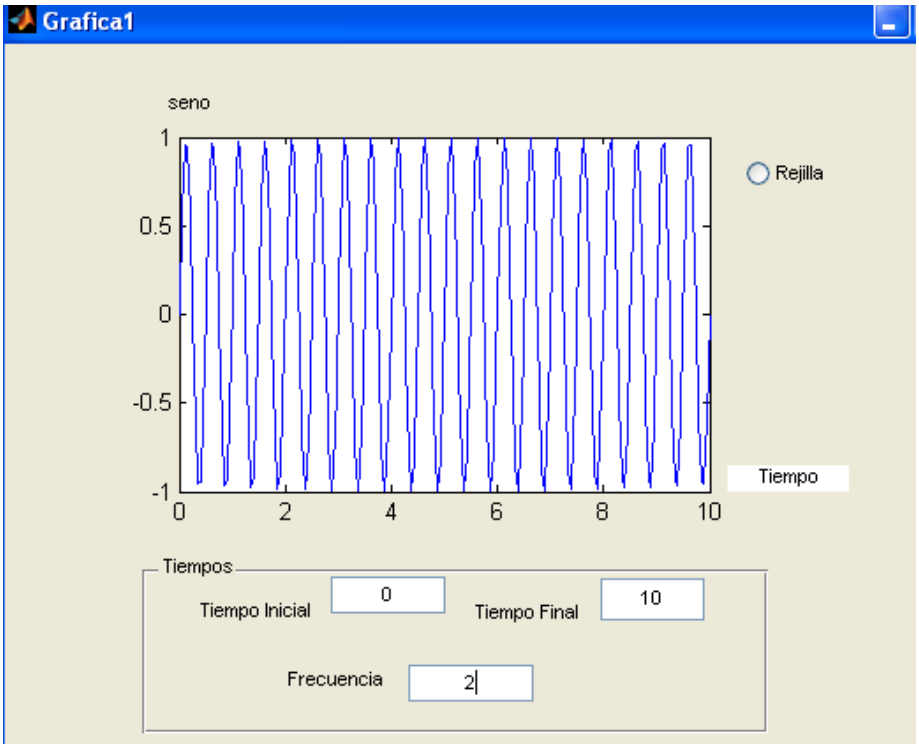
El programa genera el archivo siguiente ConvTemperatura.m :

```
% --- Executes on button press in BotonCelsius.
function BotonCelsius_Callback(hObject, eventdata, handles)
% hObject    handle to BotonCelsius (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Adicionamos
% Para leer el dato colocado en celsius
celsius1=eval(get(handles.celsius,'string'));
%Para convertir celsius a kelvin y fehrenheit
kelvin1=celsius1 + 273.15;
fahrenheit1=1.8*celsius1 + 32;
%Para escribir datos en los Edit Text
set(handles.kelvin,'string',kelvin1);
set(handles.fahrenheit,'string',fahrenheit1);

% --- Executes on button press in BotonKelvin.
function BotonKelvin_Callback(hObject, eventdata, handles)
% hObject    handle to BotonKelvin (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Adicionamos
% Para leer el dato colocado en kelvin
kelvin1=eval(get(handles.kelvin,'string'));
%Para convertir kelvin a celsius y fehrenheit
```

```
celsius1=kelvin1 - 273.15;
fahrenheit1=(kelvin1-273.15)*1.8 + 32;
%Para escribir datos en los Edit Text
set(handles.celsius,'string',celsius1);
set(handles.fahrenheit,'string',fahrenheit1);

% --- Executes on button press in BotonFarenheit.
function BotonFarenheit_Callback(hObject, eventdata, handles)
% hObject    handle to BotonFarenheit (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Adicionamos
% Para leer el dato colocado en fahrenheit
fahrenheit1=eval(get(handles.fahrenheit,'string'));
%Para convertir fahrenheit a celsius y kelvin
celsius1=(fahrenheit1-32)*5/9;
kelvin1=(fahrenheit1-32)*5/9 + 273.15;
%Para escribir datos en los Edit Text
set(handles.celsius,'string',celsius1);
set(handles.kelvin,'string',kelvin1);

% --- Executes on button press in BotonSalir.
function BotonSalir_Callback(hObject, eventdata, handles)
% hObject    handle to BotonSalir (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
close(gcbf)
```

Ejemplo: Grafica1.fig



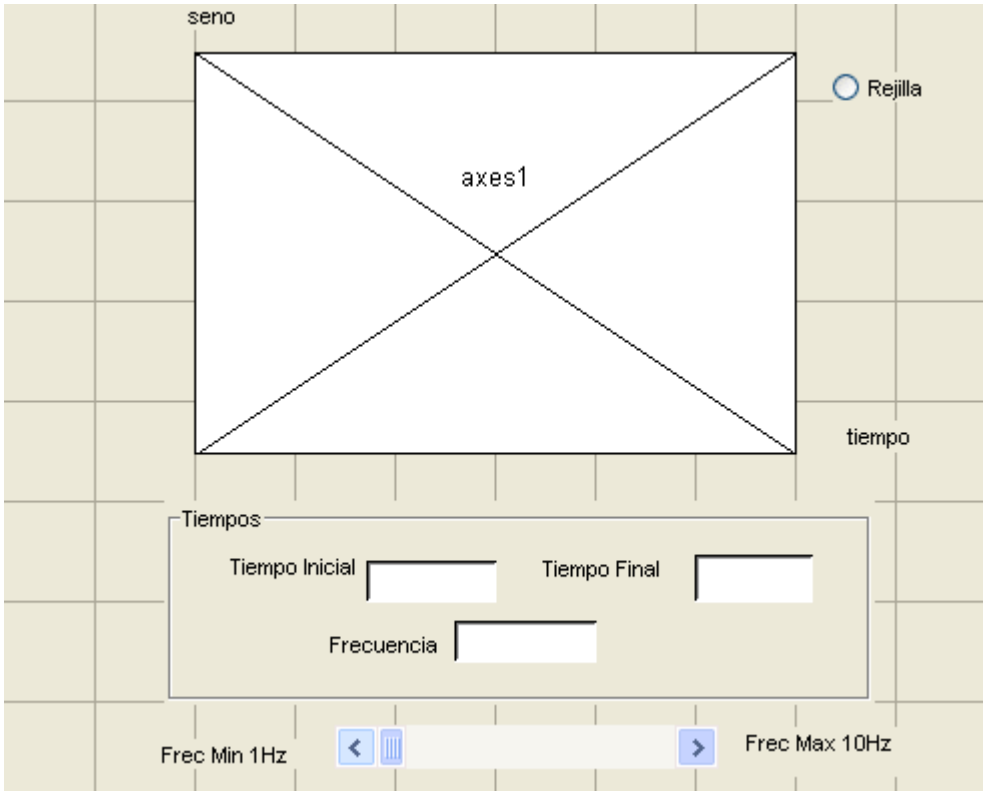


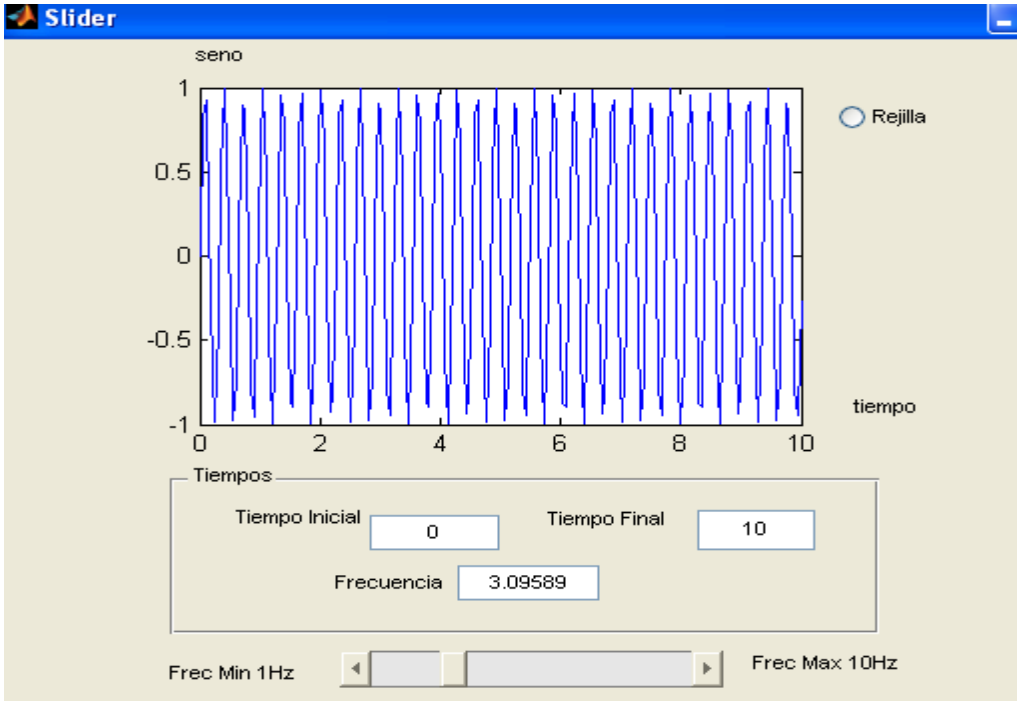
```
function Frecuencia_Callback(hObject, eventdata, handles)
% hObject    handle to Frecuencia (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of Frecuencia as text
%  str2double(get(hObject,'String')) returns contents of Frecuencia as a double

% Rangos de tiempo
t1=str2num(get(handles.TiempoInicial,'string'));
t2=str2num(get(handles.TiempoFinal,'string'));

% Vector tiempo
t=linspace(t1,t2,200);
% Valor de la frecuencia
frec=str2num(get(handles.Frecuencia,'string'));
% graficar función seno
y=sin(2*pi*frec*t);
plot(t,y);
```

Ejemplo: Slider.fig

Propiedades de la barra de deslizamiento:

Min: 1; Max: 10; Value: 5


seno
Rejilla
axes1
tiempo
Tiempos
Tiempo Inicial
Tiempo Final
Frecuencia
Frec Min 1Hz
Frec Max 10Hz


Slider
seno
Rejilla
1
0.5
0
-0.5
-1
0
2
4
6
8
10
tiempo
Tiempos
Tiempo Inicial
0
Tiempo Final
10
Frecuencia
3.09589
Frec Min 1Hz
Frec Max 10Hz

```
function Frecuencia_Callback(hObject, eventdata, handles)
% hObject    handle to Frecuencia (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of Frecuencia as text
%        str2double(get(hObject,'String')) returns contents of Frecuencia as a double

% Rangos de tiempo
t1=str2num(get(handles.TiempoInicial,'string'));
t2=str2num(get(handles.TiempoFinal,'string'));
% Vector tiempo
t=linspace(t1,t2,200);
% Valor de la frecuencia
frec=str2num(get(handles.Frecuencia,'string'));
% Barra de desplazamiento
editamin=get(handles.Barra,'Min');
editamax=get(handles.Barra,'Max');
%Chequear si el valor de frecuencia es numerico
if isnumeric(frec)&lenght(frec)==1&frec>=editamin&frec<=editamax
   set(handles.Barra,'Value',frec)
elseif frec<editamin
   set(gcbo,'string',editamin);
   set(handles.Barra,'Value',editamin);
   frec=editamin;
elseif frec>editamax
   set(gcbo,'string',editamax);
   set(handles.Barra,'value',editamax);
   frec=editamax
end
```

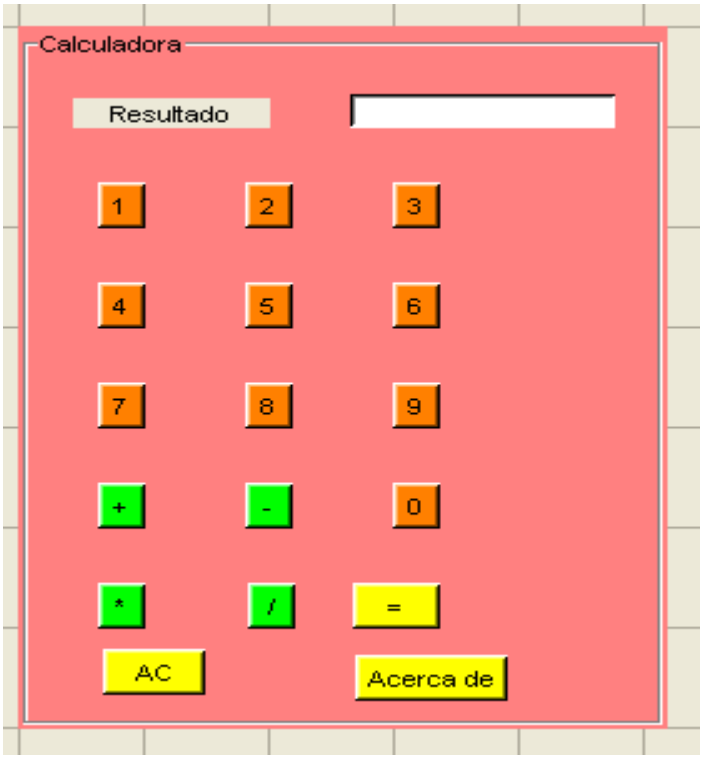
```
% graficar función seno
y=sin(2*pi*frec*t);
plot(t,y);

function Barra_Callback(hObject, eventdata, handles)
% hObject    handle to Barra (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'Value') returns position of slider
%  get(hObject,'Min') and get(hObject,'Max') to determine range of slider

set(handles.Frecuencia,'string',get(gcbo,'value'));
% Rangos de tiempo
t1=str2num(get(handles.TiempoInicial,'string'));
t2=str2num(get(handles.TiempoFinal,'string'));
% Vector tiempo
t=linspace(t1,t2,200);
frec=get(gcbo,'value');
y=sin(2*pi*frec*t);
plot(t,y);
```

Ejemplo: Calculadora.fig

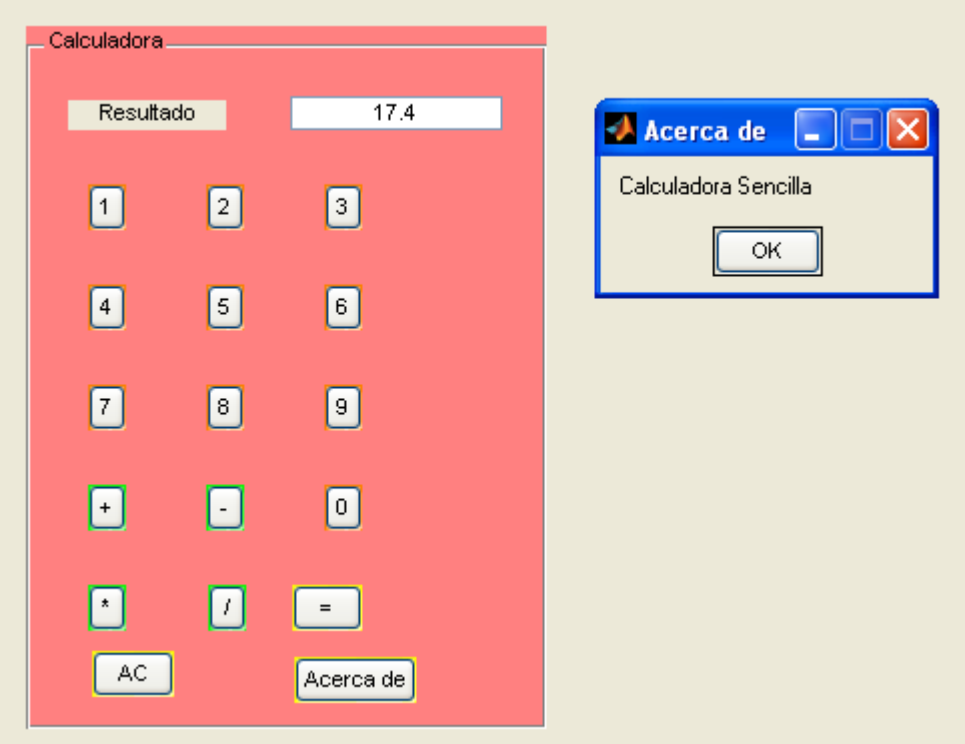
Calculadora.m

```
% --- Executes on button press in uno.
function uno_Callback(hObject, eventdata, handles)
% hObject    handle to uno (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
```

```
% handles    structure with handles and user data (see GUIDATA)

textString=get(handles.Res,'String');
textString=strcat(textString,'1');
set(handles.Res,'String',textString)

% --- Executes on button press in dos.
function dos_Callback(hObject, eventdata, handles)
% hObject    handle to dos (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
textString=get(handles.Res,'String');
textString=strcat(textString,'2');
set(handles.Res,'String',textString)
```



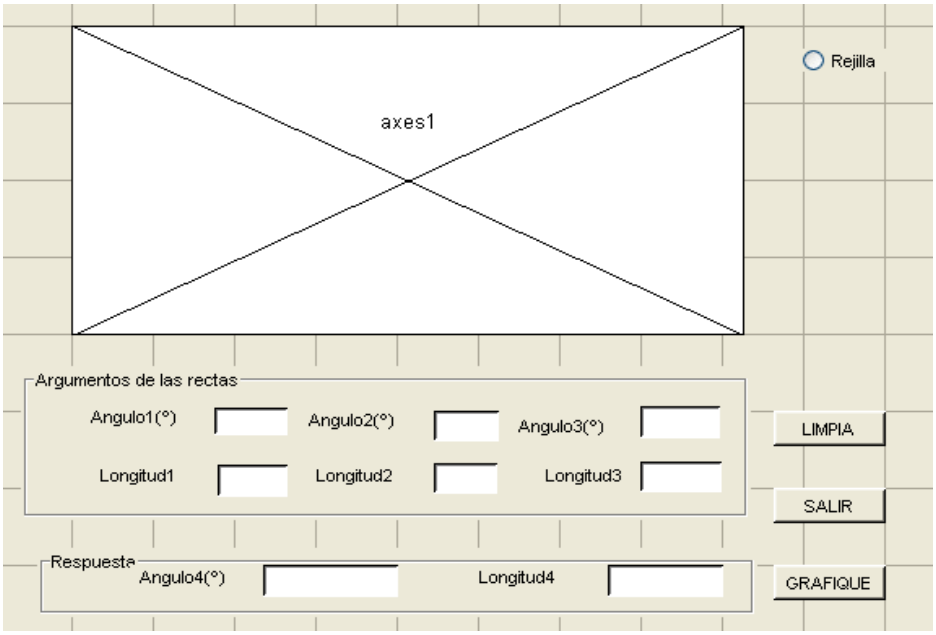
Al correr el programa se obtiene:

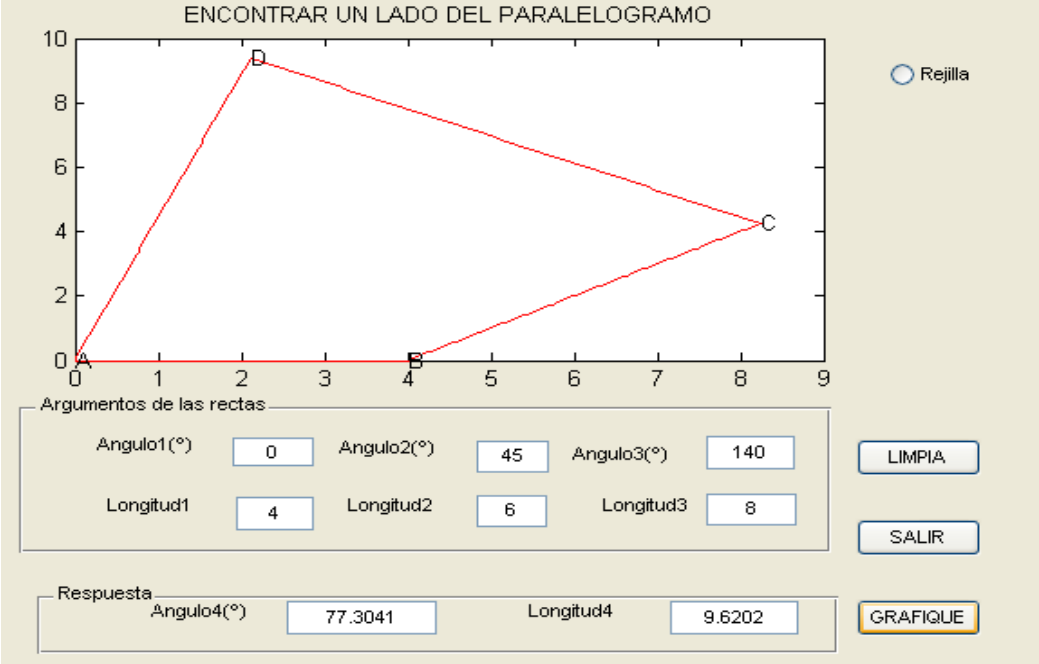



```
% --- Executes on button press in suma.
function suma_Callback(hObject, eventdata, handles)
% hObject    handle to suma (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
textString=get(handles.Res,'String');
textString=strcat(textString,'+');
set(handles.Res,'String',textString)

% --- Executes on button press in igual.
function igual_Callback(hObject, eventdata, handles)
% hObject    handle to igual (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
```

```
% handles    structure with handles and user data (see GUIDATA)
textString=get(handles.Res,'String');
textString=eval(textString,'=');
set(handles.Res,'String',textString)
% --- Executes on button press in borrar.
function borrar_Callback(hObject, eventdata, handles)
% hObject    handle to borrar (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
ini=char(' ');
set(handles.Res,'String',ini);
% --- Executes on button press in acerca.
function acerca_Callback(hObject, eventdata, handles)
% hObject    handle to acerca (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
msgbox('Calculadora Sencilla','Acerca de');
```

Ejemplo: Paralelogramo.fig



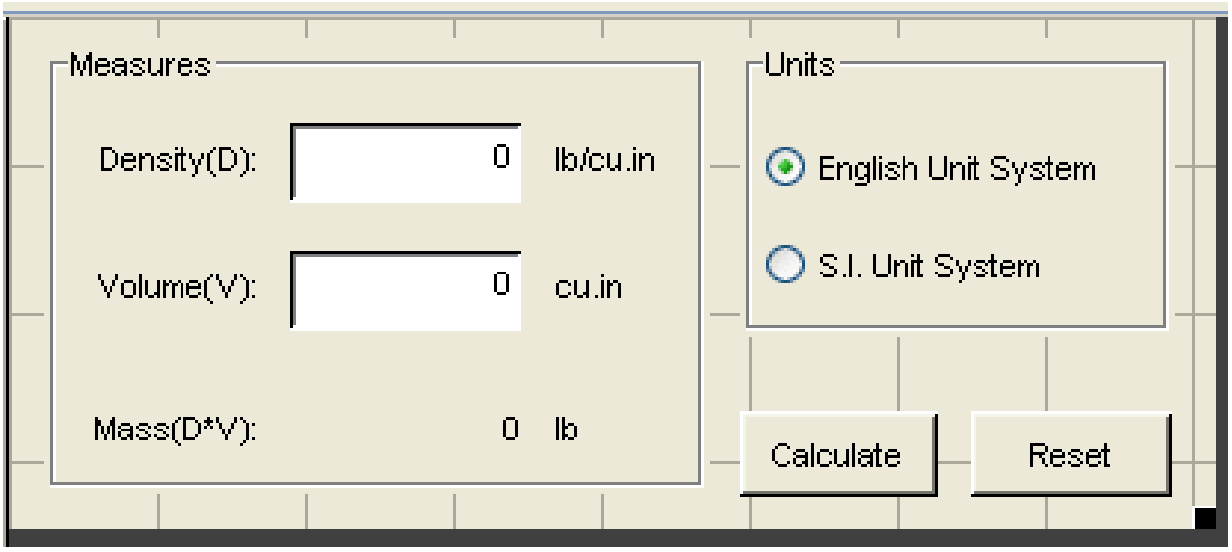


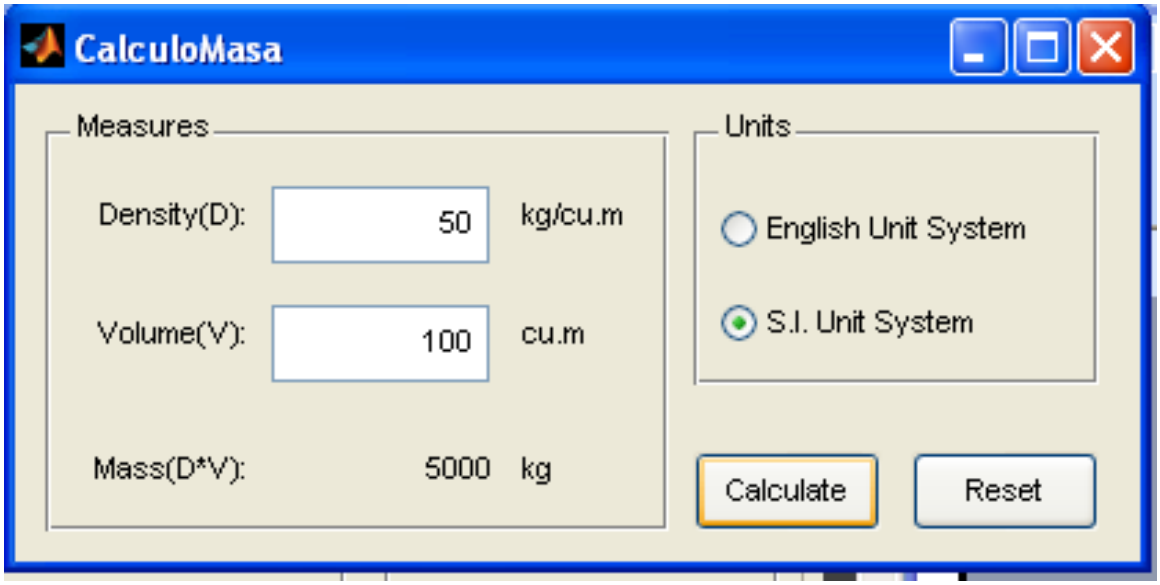

```
% --- Executes on button press in GRAFIQUE.
function GRAFIQUE_Callback(hObject, eventdata, handles)
% hObject    handle to GRAFIQUE (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Argumentos de las rectas
angulo1=str2num(get(handles.Ang1,'string'));
angulo2=str2num(get(handles.Ang2,'string'));
angulo3=str2num(get(handles.Ang3,'string'));
longitud1=str2num(get(handles.Long1,'string'));
longitud2=str2num(get(handles.Long2,'string'));
longitud3=str2num(get(handles.Long3,'string'));
% Valor del vértice inicial
x(1)=0;
y(1)=0;
```

```
%Argumentos en forma matricial
lineas=[angulo1,longitud1;angulo2,longitud2;angulo3,longitud3];
%Cálculo de las líneas
for i=1:3
    angr=lineas(i,1)*pi/180;
    m(i)=tan(angr);
    x(i+1)=x(i)+lineas(i,2)*cos(angr);
    y(i+1)=y(i)+lineas(i,2)*sin(angr);
    delta=(x(i+1)-x(i))/100;
    mx=x(i):delta:x(i+1);
    my=m(i)*(mx-x(i))+y(i);
    plot(mx,my,'r')
    vertice=['A';'B';'C';'D'];
    text(x(i),y(i),vertice(i))
    title('ENCONTRAR UN LADO DEL PARALELOGRAMO')
    hold on
end
% Argumentos de la cuarta recta
m=(y(1)-y(i+1))/(x(1)-x(i+1));
angr=atan(m);
ang=angr*180/pi
d=sqrt((y(1)-y(i+1))^2+(x(1)-x(i+1))^2)
delta=(x(1)-x(i+1))/100;
mx=x(i+1):delta:x(1);
my=m*(mx-x(i+1))+y(i+1);
plot(mx,my,'r')
text(x(i+1),y(i+1),vertice(i+1))
hold off

% Poner la respuesta de la cuarta recta
angulo=num2str(ang);
```

```
set(handles.Ang4,'string',angulo);
longitud=num2str(d);
set(handles.Long4,'string',longitud);

% --- Executes on button press in LIMPIA.
function LIMPIA_Callback(hObject, eventdata, handles)
% hObject    handle to LIMPIA (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
limpia=' ';
set(handles.Ang1,'string',limpia);
set(handles.Ang2,'string',limpia);
set(handles.Ang3,'string',limpia);
set(handles.Ang4,'string',limpia);
set(handles.Long1,'string',limpia);
set(handles.Long2,'string',limpia);
set(handles.Long3,'string',limpia);
set(handles.Long4,'string',limpia);

% --- Executes on button press in SALIR.
function SALIR_Callback(hObject, eventdata, handles)
% hObject    handle to SALIR (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
close(gcbf)
```

Ejemplo: ClaculoMasa.fig





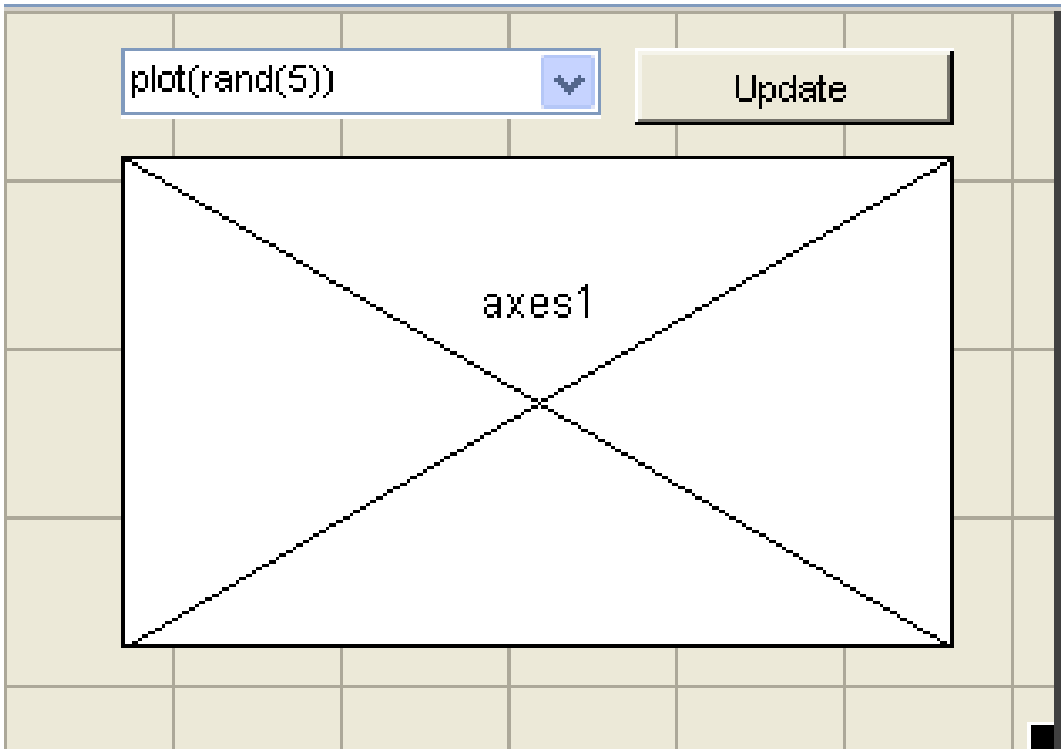
function density_Callback(hObject, eventdata, handles)

```
% hObject    handle to density (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of density as text
%        str2double(get(hObject,'String')) returns contents of density as a double

density = str2double(get(hObject, 'String'));
if isnan(density)
    set(hObject, 'String', 0);
    errordlg('Input must be a number','Error');
end

% Save the new density value
handles.metricdata.density = density;
guidata(hObject,handles)

function volume_Callback(hObject, eventdata, handles)
% hObject    handle to volume (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of volume as text
%        str2double(get(hObject,'String')) returns contents of volume as a double

volume = str2double(get(hObject, 'String'));
if isnan(volume)
    set(hObject, 'String', 0);
    errordlg('Input must be a number','Error');
end
```

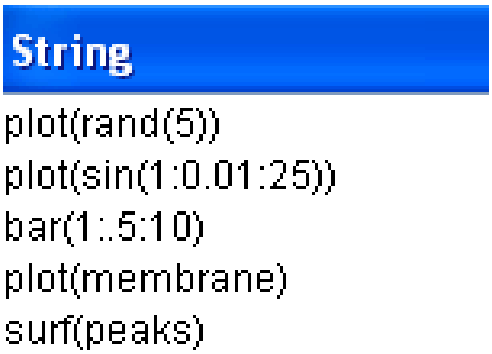
```
% Save the new volume value
handles.metricdata.volume = volume;
guidata(hObject,handles)

function calculate_Callback(hObject, eventdata, handles)
% hObject    handle to calculate (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

mass = handles.metricdata.density * handles.metricdata.volume;
set(handles.mass, 'String', mass);

% --- Executes on button press in reset.
function reset_Callback(hObject, eventdata, handles)
% hObject    handle to reset (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

initialize_gui(gcbf, handles, true);

% --------------------------------------------------------------------
function unitgroup_SelectionChangeFcn(hObject, eventdata, handles)
% hObject    handle to unitgroup (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

if (hObject == handles.english)
    set(handles.text4, 'String', 'lb/cu.in');
    set(handles.text5, 'String', 'cu.in');
    set(handles.text6, 'String', 'lb');
```

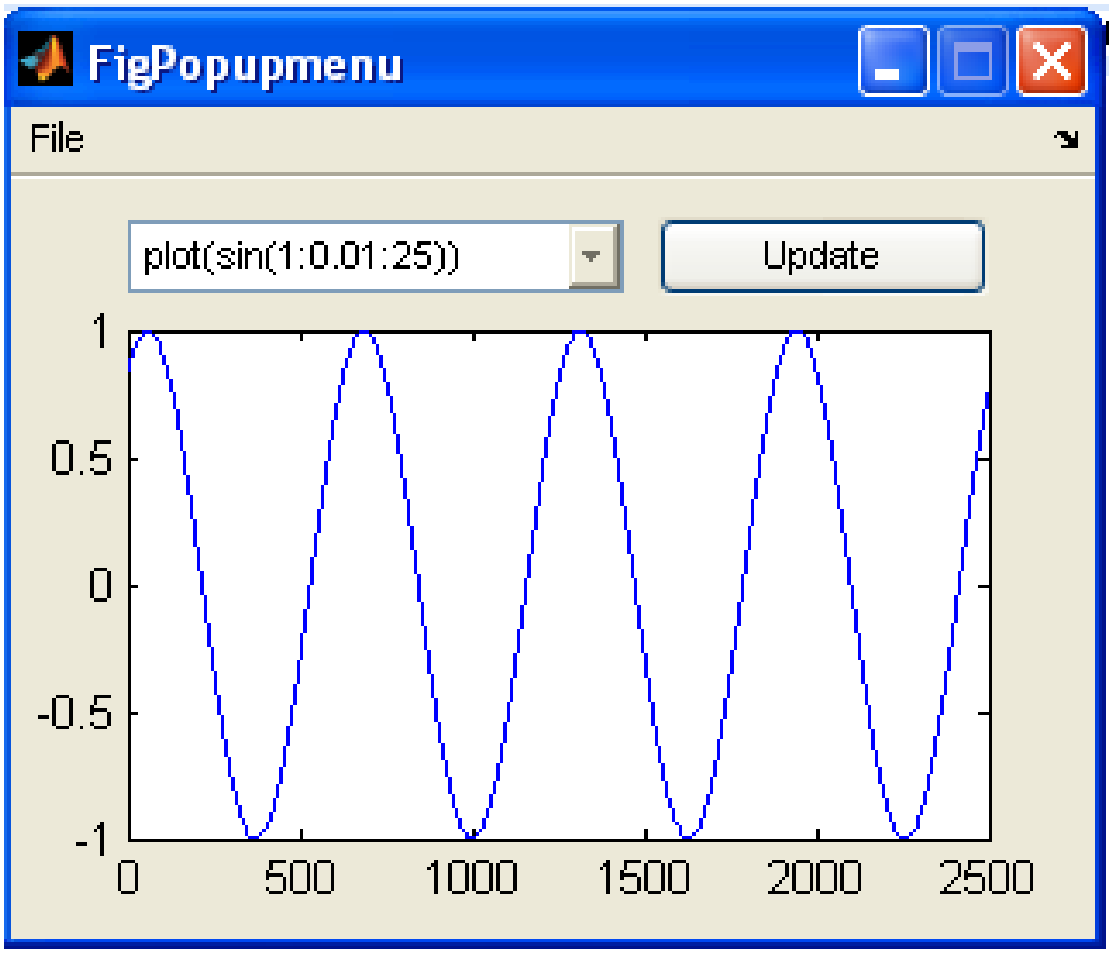
```
else
    set(handles.text4, 'String', 'kg/cu.m');
    set(handles.text5, 'String', 'cu.m');
    set(handles.text6, 'String', 'kg');
end

% --------------------------------------------------------------------
function initialize_gui(fig_handle, handles, isreset)
% If the metricdata field is present and the reset flag is false, it means
% we are we are just re-initializing a GUI by calling it from the cmd line
% while it is up. So, bail out as we dont want to reset the data.
if isfield(handles, 'metricdata') && ~isreset
    return;
end

handles.metricdata.density = 0;
handles.metricdata.volume  = 0;

set(handles.density, 'String', handles.metricdata.density);
set(handles.volume,  'String', handles.metricdata.volume);
set(handles.mass, 'String', 0);

set(handles.unitgroup, 'SelectedObject', handles.english);

set(handles.text4, 'String', 'lb/cu.in');
set(handles.text5, 'String', 'cu.in');
set(handles.text6, 'String', 'lb');

% Update handles structure
guidata(handles.figure1, handles);
```

Ejemplo: FigPopupmenu



En el inspector de propiedades en el String del popmenu se debe escribir:

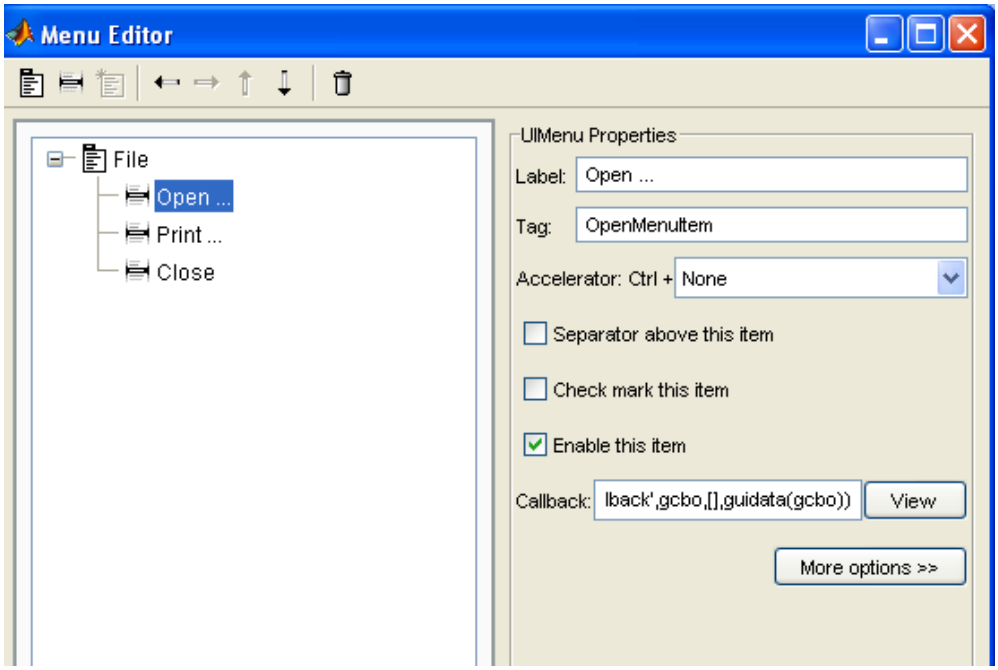


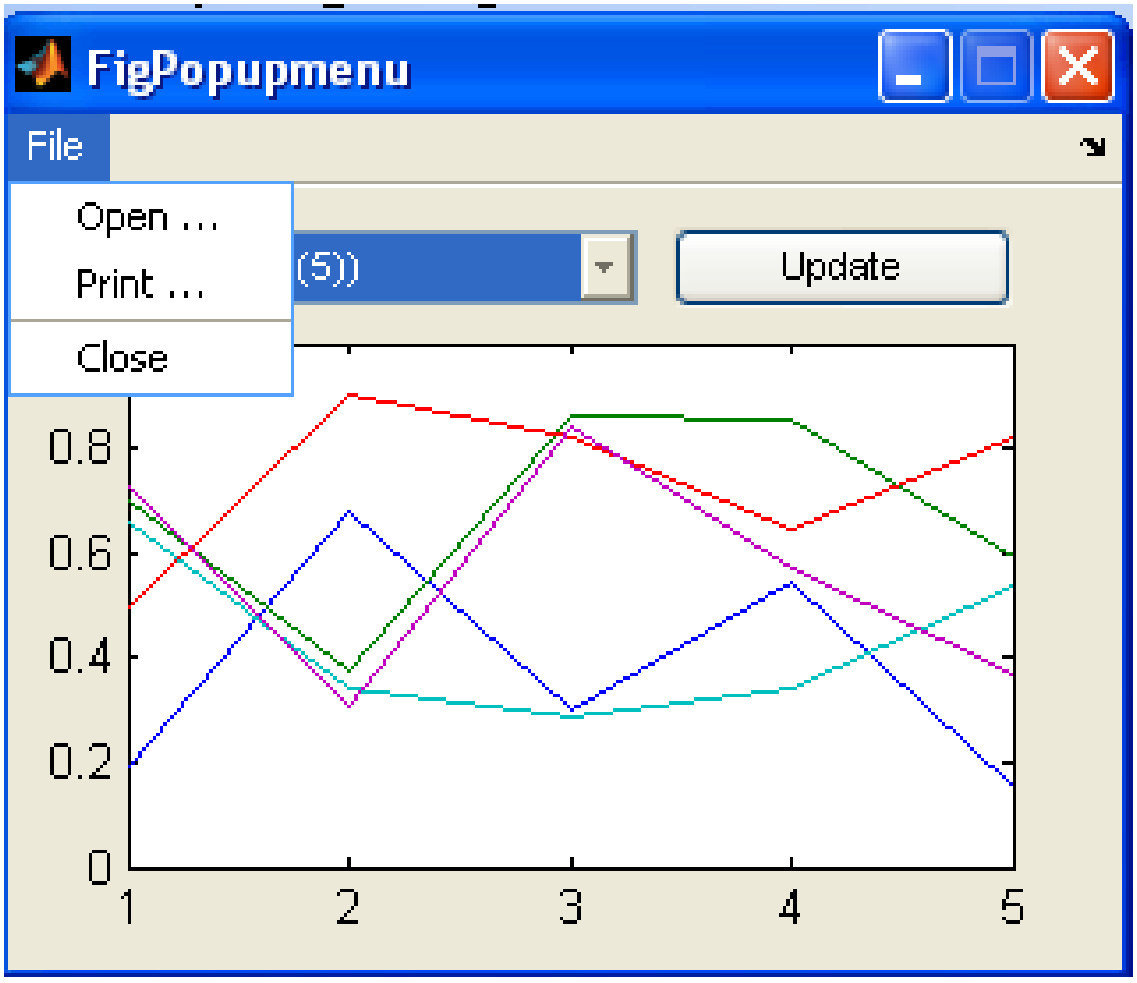



```
function FigPopupmenu_OpeningFcn(hObject, eventdata, handles, varargin)
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);

% This sets up the initial plot - only do when we are invisible
% so window can get raised using FigPopupmenu.
if strcmp(get(hObject,'Visible'),'off')
    plot(rand(5));
end
```

```
function pushbutton1_Callback(hObject, eventdata, handles)
axes(handles.axes1);
cla;
popup_sel_index = get(handles.popupmenu1, 'Value');
switch popup_sel_index
   case 1
      plot(rand(5));
   case 2
      plot(sin(1:0.01:25.99));
   case 3
      bar(1:.5:10);
   case 4
      plot(membrane);
   case 5
      surf(peaks);
end
```

MENU EDITOR:

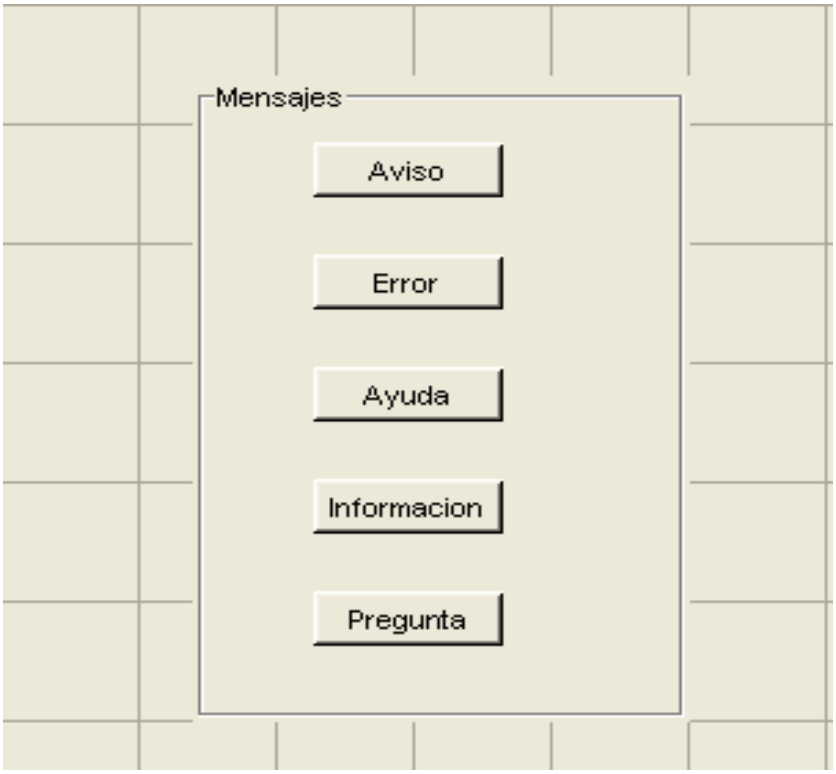





```
% --------------------------------------------------------------------
function FileMenu_Callback(hObject, eventdata, handles)
% hObject    handle to FileMenu (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% --------------------------------------------------------------------
function OpenMenuItem_Callback(hObject, eventdata, handles)
file = uigetfile('*.fig');
if ~isequal(file, 0)
```

```
    open(file);
end
% --------------------------------------------------------------------
function PrintMenuItem_Callback(hObject, eventdata, handles)
printdlg(handles.figure1)
% --------------------------------------------------------------------
function CloseMenuItem_Callback(hObject, eventdata, handles)
selection = questdlg(['Close ' get(handles.figure1,'Name') '?'],...
                     ['Close ' get(handles.figure1,'Name') '...'],...
                     'Yes','No','Yes');
if strcmp(selection,'No')
    return;
end
delete(handles.figure1)
```

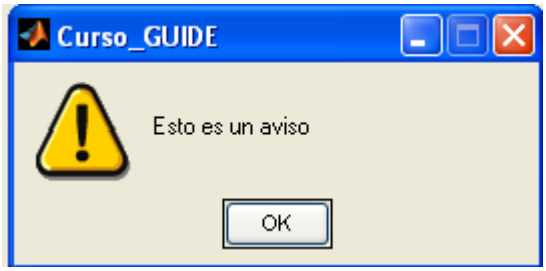
Ejemplo: Mensajes.fig



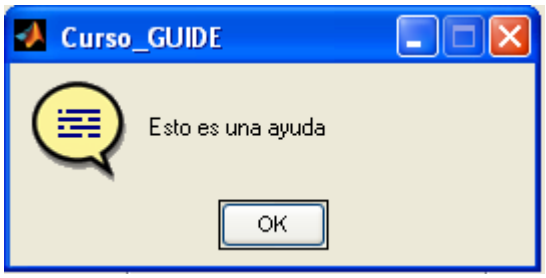
Adicionar a los Callbacks:

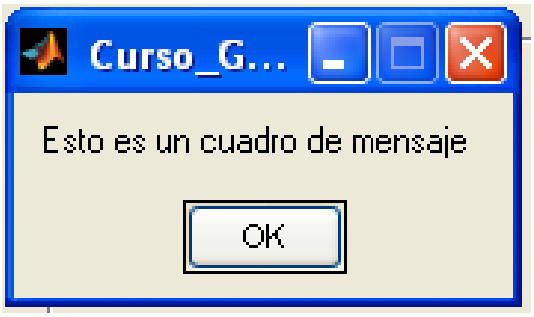
```
% --- Executes on button press in Aviso.
function Aviso_Callback(hObject, eventdata, handles)
warndlg('Esto es un aviso','Curso_GUIDE');

% --- Executes on button press in Error.
function Error_Callback(hObject, eventdata, handles)
errordlg('Esto es un mensaje de error',' Curso_GUIDE ');

% --- Executes on button press in Ayuda.
function Ayuda_Callback(hObject, eventdata, handles)
helpdlg('Esto es una ayuda',' Curso_GUIDE ');

% --- Executes on button press in Informacion.
function Informacion_Callback(hObject, eventdata, handles)
msgbox('Esto es un cuadro de mensaje',' Curso_GUIDE ');

% --- Executes on button press in Pregunta.
function Pregunta_Callback(hObject, eventdata, handles)
questdlg('Esto es una pregunta',' Curso_GUIDE ');
```

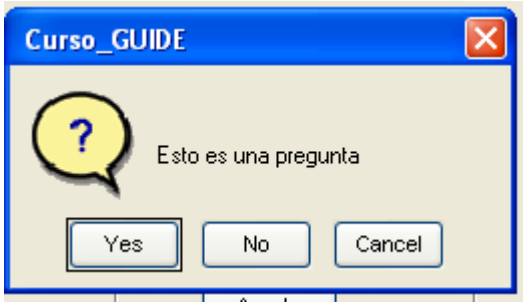
Se obtienen a ejecutar los botones las siguientes respuestas:


Curso_GUIDE
Esto es un aviso
OK


Curso_GUIDE
Esto es un mensaje de error
OK


Curso_GUIDE
Esto es una ayuda
OK


Curso_G...
Esto es un cuadro de mensaje
OK


Curso_GUIDE
Esto es una pregunta
Yes
No
Cancel

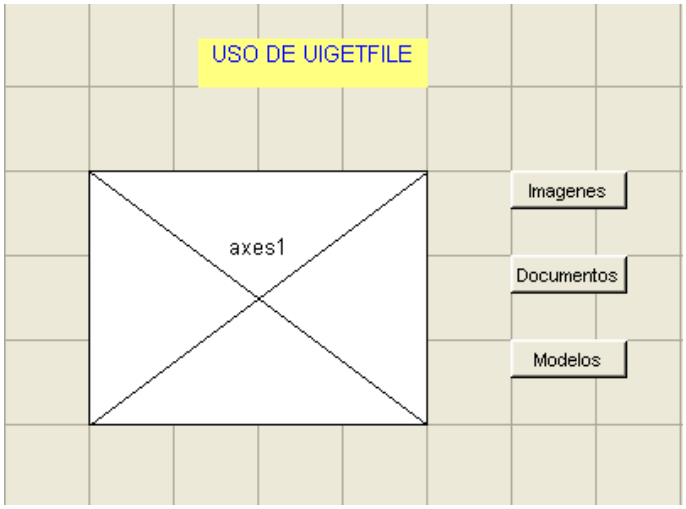
Para el caso especial de las preguntas podemos ejecutar o no sentencias dependiendo de la respuesta escogida. Por ejemplo, si deseamos salir o no del programa, se tiene[1]:

```
ans=questdlg('¿Desea salir del programa?','SALIR','Si','No','No');
if strcmp(ans,'No')
return;
end
clear,clc,close all
```

La función strcmp compara dos *strings* y si son iguales retorna el valor 1 (*true*). *Clear* elimina todas los valores de workspace, *clc* limpia la pantalla y *close all* cierra todos los Guide. Nótese que la secuencia 'Si','No','No' termina en 'No'; con esto se logra que la parte *No* del cuadro de pregunta esté resaltado. Si terminara en 'Si', la parte *Si* del cuadro de pregunta se resaltaría.

[1] **Manual de Interfaz Gráfica de Usuario en Matlab**
Por: Diego Orlando Barragán Guerrero

Ejemplo: Archivos.fig (uso de uigetfile)



Adicionar en los Callbacks:

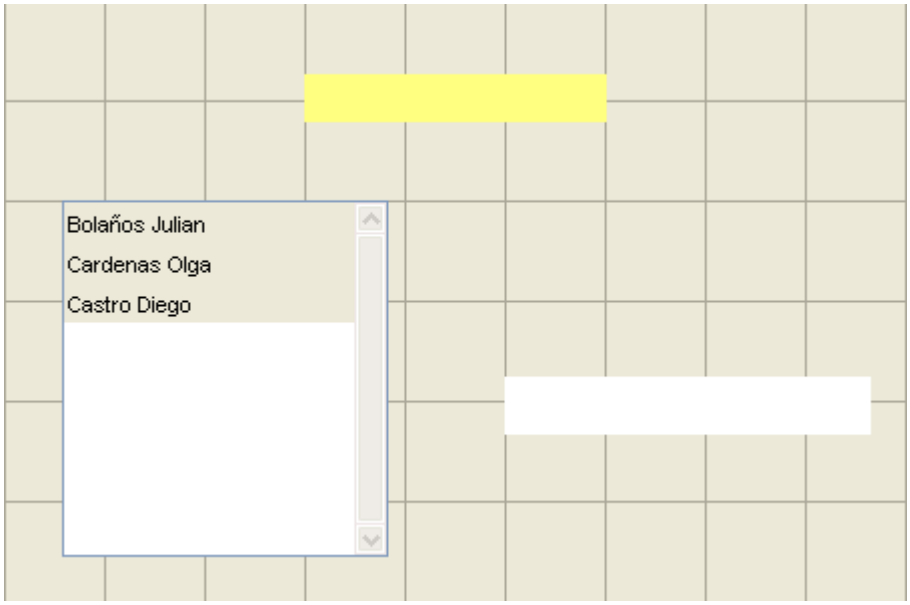
```
function IMAGEN_Callback(hObject, eventdata, handles)
[FileName Path]=uigetfile({'*.jpg;*.bmp'},'Abrir Imagen');
if isequal(FileName,0)
return
else
a=imread(strcat(Path,FileName));
imshow(a);
end
handles.direccion=strcat(Path,FileName);
guidata(hObject,handles)
```

Como se puede observar, si el usuario presiona *cancelar* el programa no ejecuta ningún proceso. La función *imread* lee una imagen en Matlab e *imshow* la presenta.

Este programa tendrá la siguiente presentación:


```
% --- Executes on button press in DOC.
function DOC_Callback(hObject, eventdata, handles)
[FileName Path]=uigetfile({'*.doc;*.xls'},'Abrir documento');
if isequal(FileName,0)
return
else
winopen(strcat(Path,FileName));
end

% --- Executes on button press in PROGRAMA.
function PROGRAMA_Callback(hObject, eventdata, handles)
[FileName Path]=uigetfile({'*.mdl'},'Abrir archivo');
if isequal(FileName,0)
return
else
open_system(strcat(Path,FileName))
end
```

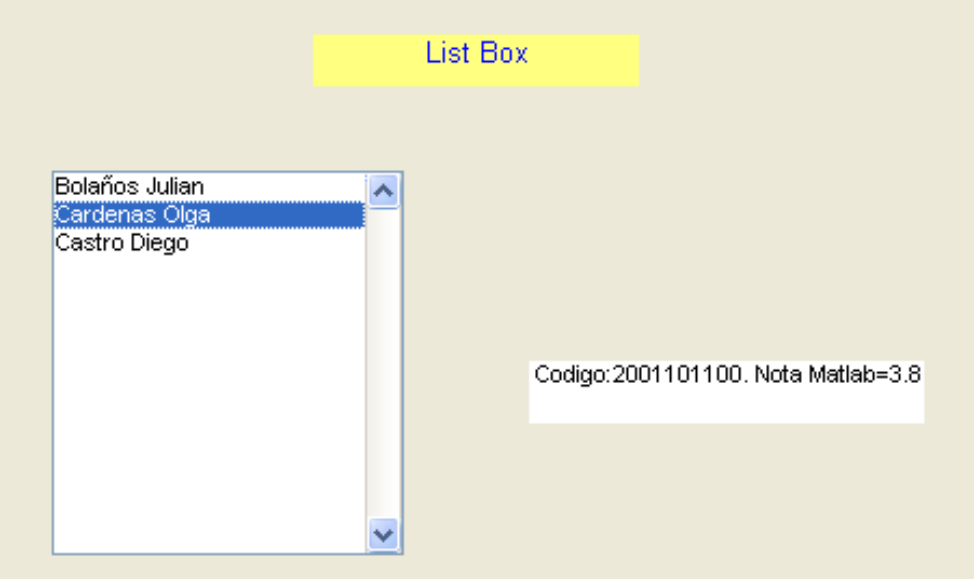
Ejemplo: Listbox.fig



Adicionamos en los Callbacks:

```
% --- Executes just before Listbox is made visible.
function Listbox_OpeningFcn(hObject, eventdata, handles, varargin)
set(handles.text1,'string','List Box');
set(handles.text2,'string',...
['Codigo:2005202275. Nota Matlab=3.5']);
% Choose default command line output for Listbox
handles.output = hObject;
% Update handles structure
guidata(hObject, handles);

% --- Executes on selection change in listbox1.
function listbox1_Callback(hObject, eventdata, handles)
% Hints: contents = get(hObject,'String') returns listbox1 contents as cell array
```

```
%       contents{get(hObject,'Value')} returns selected item from listbox1
n=get(hObject,'Value');
gos=get(hObject,'String');
switch n
case 1
set(handles.text2,'string',...
['Codigo:2005202275. Nota Matlab=3.5']);
case 2
set(handles.text2,'string',...
['Codigo:2001101100. Nota Matlab=3.8']);
case 3
set(handles.text2,'string',...
['Codigo:2006264188. Nota Matlab=4.5']);
end
guidata(hObject,handles);
```



5. GUI - SIMULINK

Ejemplo: Gui_simulik.fig (Pop up Menu)

Comandos utilizados:

find_system: comprueba si existe el programa en simulink

find_System('Type','Nombre_simulink')

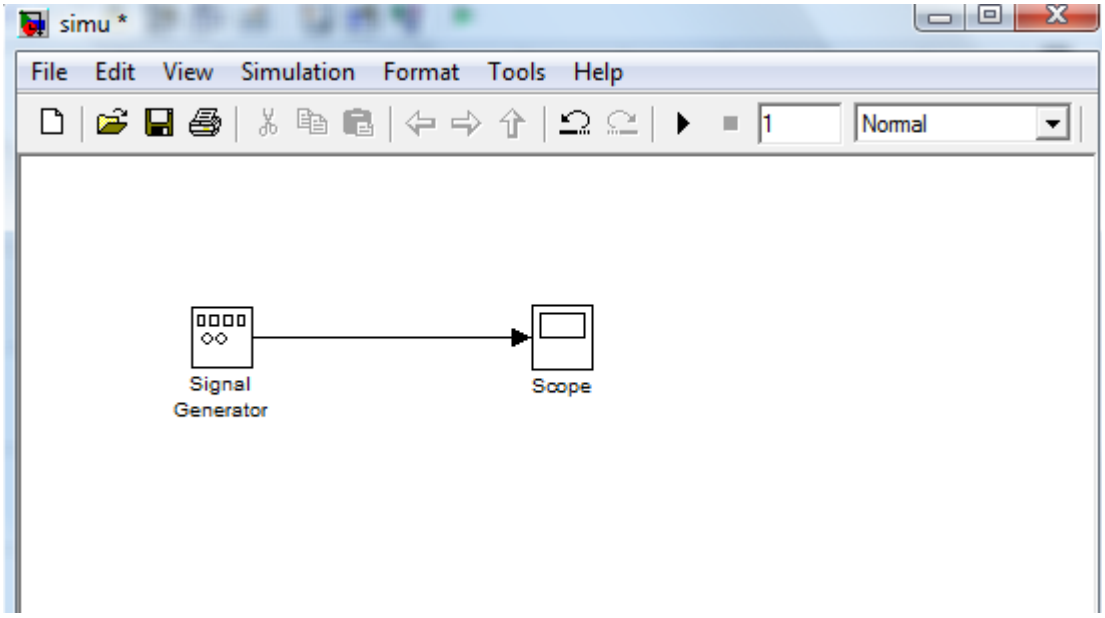
open_System: Abre el programa Simulink

open_system('Nombre_simulink')

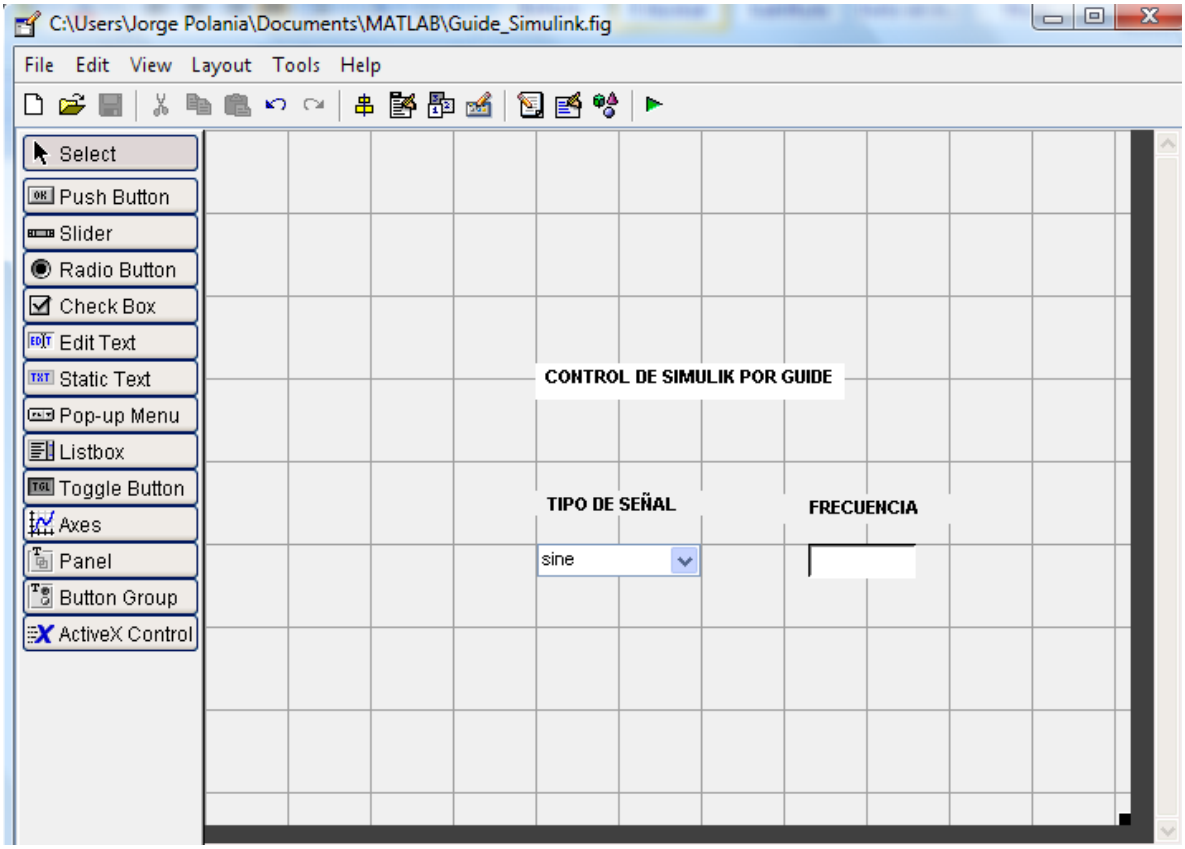
set_param: Para escribir en los bloques de simulik

set_param('Nombre_simulink/'Nombre_bloque')

PROGRAMA EN SIMULINK:



PROGRAMA EN GUIDE:

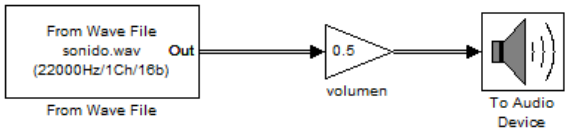


PROGRAMA EN MATLAB:

```
function Guide_Simulink_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 Guide_Simulink (see VARARGIN)

% Choose default command line output for Guide_Simulink
% Update handles structure
handles.output = hObject;
guidata(hObject, handles);
%--------------------------------------------------
find_system('Type','simu');
open_system('simu');
set_param('simu/Signal Generator','Waveform','sine','frequency','5');
set_param('simu','Solver','Ode23','StopTime','1');
set_param(gcs,'SimulationCommand','start');
%--------------------------------------------------
% UIWAIT makes Guide_Simulink wait for user response (see UIRESUME)
% uiwait(handles.figure1);


function wave_Callback(hObject, eventdata, handles)
% hObject    handle to wave (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see
% GUIDATA)
```

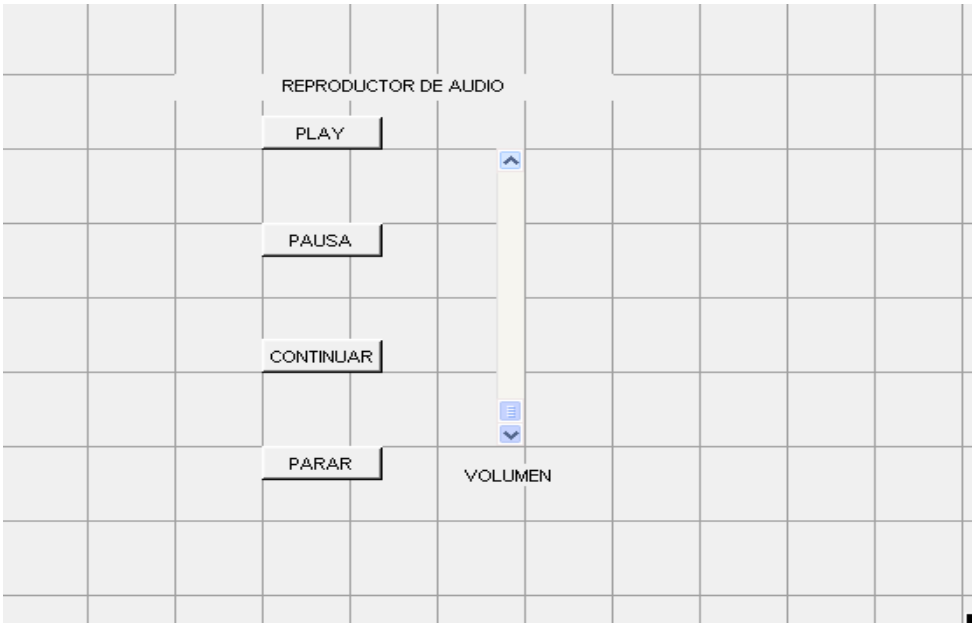
```
%-------------------------------------------------------------
onda=get(hObject,'Value');
if onda==1
   set_param('simu/Signal Generator','Waveform','sine');
   set_param(gcs,'simulationCommand','Start');
elseif onda==2
   set_param('simu/Signal Generator','Waveform','square');
   set_param(gcs,'simulationCommand','Start');
elseif onda==3
   set_param('simu/Signal Generator','Waveform','sawtooth');
   set_param(gcs,'simulationCommand','Start');
else
   set_param('simu/Signal Generator','Waveform','random');
   set_param(gcs,'simulationCommand','Start');
end
%-------------------------------------------------------------

function edit1_Callback(hObject, eventdata, handles)
% hObject    handle to edit1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit1 as text
%        str2double(get(hObject,'String')) returns contents of edit1 as a double

%-------------------------------------------------------------------
f=get(hObject,'string');
set_param('simu/Signal Generator','Frequency',f);
set_param('Simulationcommand','Start');
%-------------------------------------------------------------------
```

Ejemplo: Reproducir sonido música.mdl



Los bloques se encuentran en el Toolbox de Signal Processing, como Source y Sink

audio.fig (archivos.wav)



El Slider debe estar en Max= 10.0 y Min=0.0 para que se pueda escuchar el cambio de volumen.

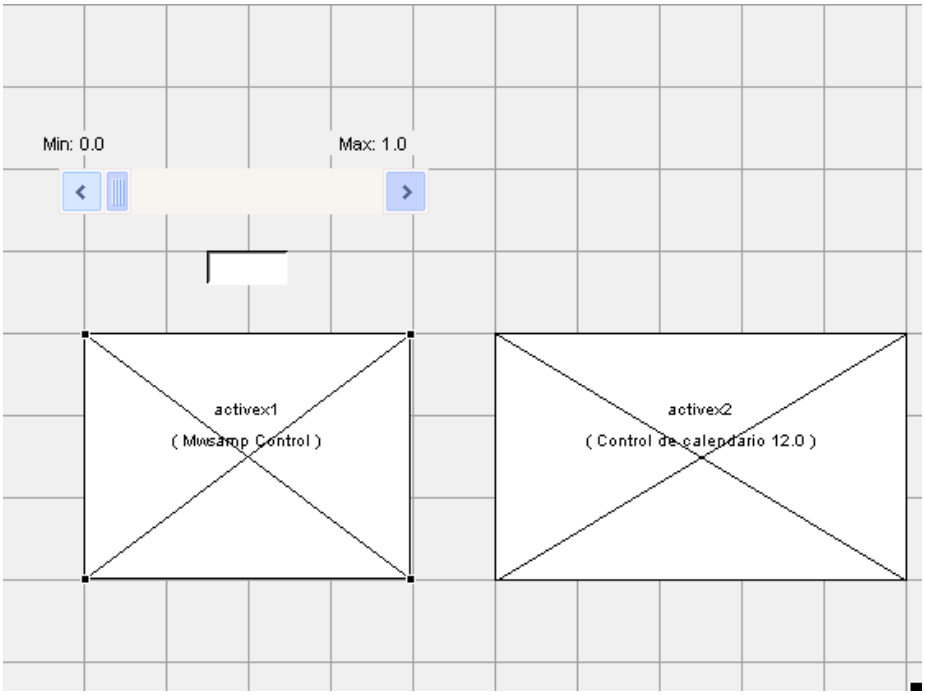
Audio.m

```
function audio_OpeningFcn(hObject, eventdata, handles, varargin)
%--------------------------------------------------------------------
set(handles.volumen,'Value',0.5);
find_system('Name','musica');

open_system('musica');
set_param('musica/volumen','Gain','0.5');
set_param('musica/From Wave File','FileName','sonido.wav');


function play_Callback(hObject, eventdata, handles)
%-----------------------------------------------------------
set_param('musica/From Wave File','FileName','sonido.wav');
set_param(gcs,'SimulationCommand','Start');

function pausa_Callback(hObject, eventdata, handles)
%----------------------------------------
set_param(gcs,'SimulationCommand','Pause')

function continuar_Callback(hObject, eventdata, handles)
%-----------------------------------------------
set_param(gcs,'SimulationCommand','Continue')

function parar_Callback(hObject, eventdata, handles)
%-------------------------------------------------------
```

```
set_param(gcs,'SimulationCommand','Stop')

function volumen_Callback(hObject, eventdata, handles)
%--------------------------------------------------------------------
vol=get(hObject,'Value');
set_param('musica/volumen','Gain',num2str(vol));
```

Ejemplo: ActiveX

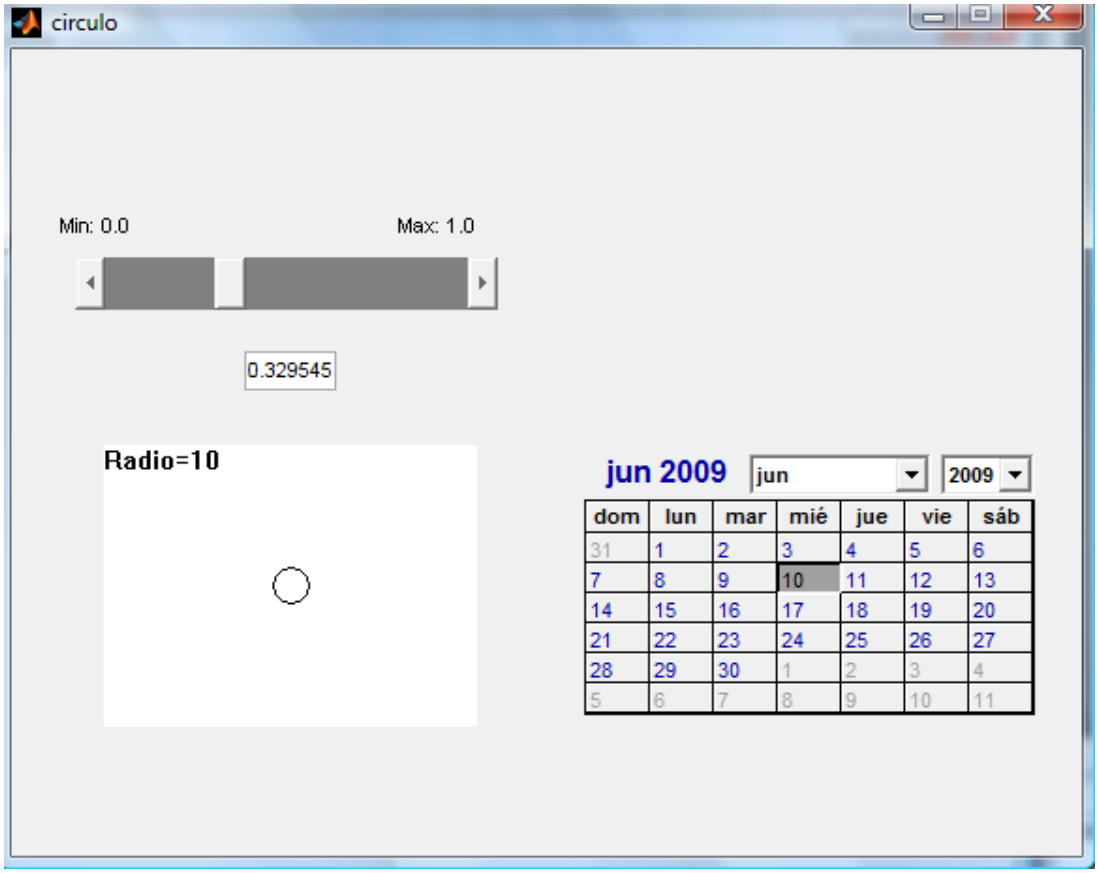
Circulo.fig (mwsamp control)



Circulo.m

```
function circulo_OpeningFcn(hObject, eventdata, handles, varargin)
%---------------------------------
a=num2str(handles.activex1.Radius);
```

```
handles.activex1.Label=['Radio=' a];
%-----------------------------------

function activex1_Click(hObject, eventdata, handles)
% hObject    handle to activex1 (see GCBO)
% eventdata  structure with parameters passed to COM event listener
% handles    structure with handles and user data (see GUIDATA)

%-------------------------------------
 c=get(handles.slider1,'Value');
RadioInicial=(1/c)*handles.activex1.radius;
handles.activex1.Radius=RadioInicial;
handles.activex1.Label=['Radio=',num2str(handles.activex1.Radius)];
refresh(handles.figure1);

% --- Executes on slider movement.
function slider1_Callback(hObject, eventdata, handles)
%-----------------------------------------------------
c=get(hObject,'Value');
set(handles.edit1,'string',c);
RadioInicial=handles.activex1.radius;
handles.activex1.Radius=c*RadioInicial;
handles.activex1.Label=['Radio=',num2str(handles.activex1.Radius)];
refresh(handles.figure1);
```



En la figura también se ha utilizado para tener el calendario, el ActiveX Control de calendario 12.0

Compilar GUI con ActiveX

mcc -m nombregui -a nombregui_activex1