Sandwich Check Box Demo (Version 1.1)

Created by Dr. C. S. Tritt
Last revised: January 16, 2007

Background and Equations

None required.

Operations Description

Demonstrates the use of check boxes. Message changes in response to selection. Choices are sandwich options are lettuce, tomato and onion. The message updates as choices are made. Clicking the place order button displays the appropriate message in the Matlab command window via a function in a separate file. Clicking the clear button clears any selected check boxes and the message.

User Interface Description

Three check boxes (lettuceBox, tomatoBox and onionBox).

A “Order” button (orderButton).

A “Clear” button (clearButton)

A static text control (labelText) to label the message.

A static text control (orderText) that is the message.

Algorithms

_OpeningFcn

Create and initialize check box state property variables (lettuceState, tomatoState, onionState) in handles structure (allowed values are 0 and 1).

Call updateMessage to create initial message, display it and create and initialize the message text in the handles structure. This function also updates the central gui data structure.

lettuceBox_Callback, *tomatoBox_Callback* and *onionBox_Callback*

These functions are identical except that the process different ingredients.

Get the check box state and assign it to the corresponding handle property.

Call the updateMessage function to update the message text, display it and update the central gui data structure.

orderButton_Callback

Call the separate (independent) displayOrder function to display the order in the command window. Note that this function could be modified to send the message across the internet or to a piece of hardware.

clearButton_Callback

```
For each check box:

If  the State is 1 (true, selected)
    Reset the Value property of the corresponding check box to 0 (the default Min).
    Change the handles state property to 0

Call the updateMessage function to update the message text, display it and update the
central gui data structure.
```

updateMessage(hObject, handles)

```
Create and initialize a blank message.

For each ingredient:

If the state property is 1
    Add the corresponding ingredient text to the message.

If message is still blank
    Make message ‘Plain’

Store message as a handle property (messageStr).

Update the central gui data structure (guidata(hObject, handles)).
```

Source Code

```
function varargout = sandwichDemo(varargin)
% sandwichDemo M-file for sandwichDemo.fig
%
% See SandwichCheckBoxExample.doc for more information.
%
% Created by Dr. C. S. Tritt
% Last revised 1/18/07
%
% See also: GUIDE, GUIDATA, GUIHANDLES

% Copyright 2002-2003 The MathWorks, Inc.

% Edit the above text to modify the response to help sandwichDemo

% Last Modified by GUIDE v2.5 17-Jan-2007 17:03:37

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @sandwichDemo_OpeningFcn, ...
                   'gui_OutputFcn',  @sandwichDemo_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT


% --- Executes just before sandwichDemo is made visible.
function sandwichDemo_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 sandwichDemo (see VARARGIN)

% Choose default command line output for sandwichDemo
handles.output = hObject;

% Added Code: Create and initialize check box state and message
% property variables.

handles.lettuceState = 0;
handles.tomatoState = 0;
handles.onionState = 0;
```

```
% Call function to update and redisplay the message.
% Note: This function will update guidata!
updateMessage(hObject, handles);

% UIWAIT makes sandwichDemo wait for user response (see UIRESUME)
% uiwait(handles.figure1);


% --- Outputs from this function are returned to the command line.
function varargout = sandwichDemo_OutputFcn(hObject, eventdata,
handles)
% varargout  cell array for returning output args (see VARARGOUT);
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout{1} = handles.output;


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

% Update lettuce state variable.
handles.lettuceState = get(hObject,'Value');

% Call function to update and redisplay the message.
% Note: This function will update guidata!
updateMessage(hObject, handles);

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

% Update tomato state variable.
handles.tomatoState = get(hObject,'Value');

% Call function to update and redisplay the message.
% Note: This function will update guidata!
updateMessage(hObject, handles);


% --- Executes on button press in checkbox3.
function onionBox_Callback(hObject, eventdata, handles)
% hObject    handle to checkbox3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Update onion state variable.
handles.onionState = get(hObject,'Value');

% Call function to update and redisplay the message.
% Note: This function will update guidata!
updateMessage(hObject, handles);


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

% Display the order message.
displayOrder(handles.messageStr);

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

if handles.lettuceState
   set(handles.lettuceBox, 'Value', 0); % 0 is default Min property.
   handles.lettuceState = 0;
end

if handles.tomatoState
   set(handles.tomatoBox, 'Value', 0); % 0 is default Min property.
   handles.tomatoState = 0;
end

if handles.onionState
   set(handles.onionBox, 'Value', 0); % 0 is default Min property.
   handles.onionState = 0;
end

% Call function to update and redisplay the message.
% Note: This function will update guidata!
updateMessage(hObject, handles);

function updateMessage(hObject, handles)
%
% This function upates the message text based on the state of the check
% boxes.
%

message = ''; % Start with an empty message.

% Add selected ingrendents.

if handles.lettuceState
   message = [message 'Lettuce '];
end

if handles.tomatoState
   message = [message 'Tomato '];
end

if handles.onionState
```

```
   message = [message 'Onion '];
end

if strcmp(message, '')
   message = 'Plain';
end

% Store and display the message.

handles.messageStr = message;
set(handles.orderText, 'String', message);

% Update handles structure
guidata(hObject, handles);
```