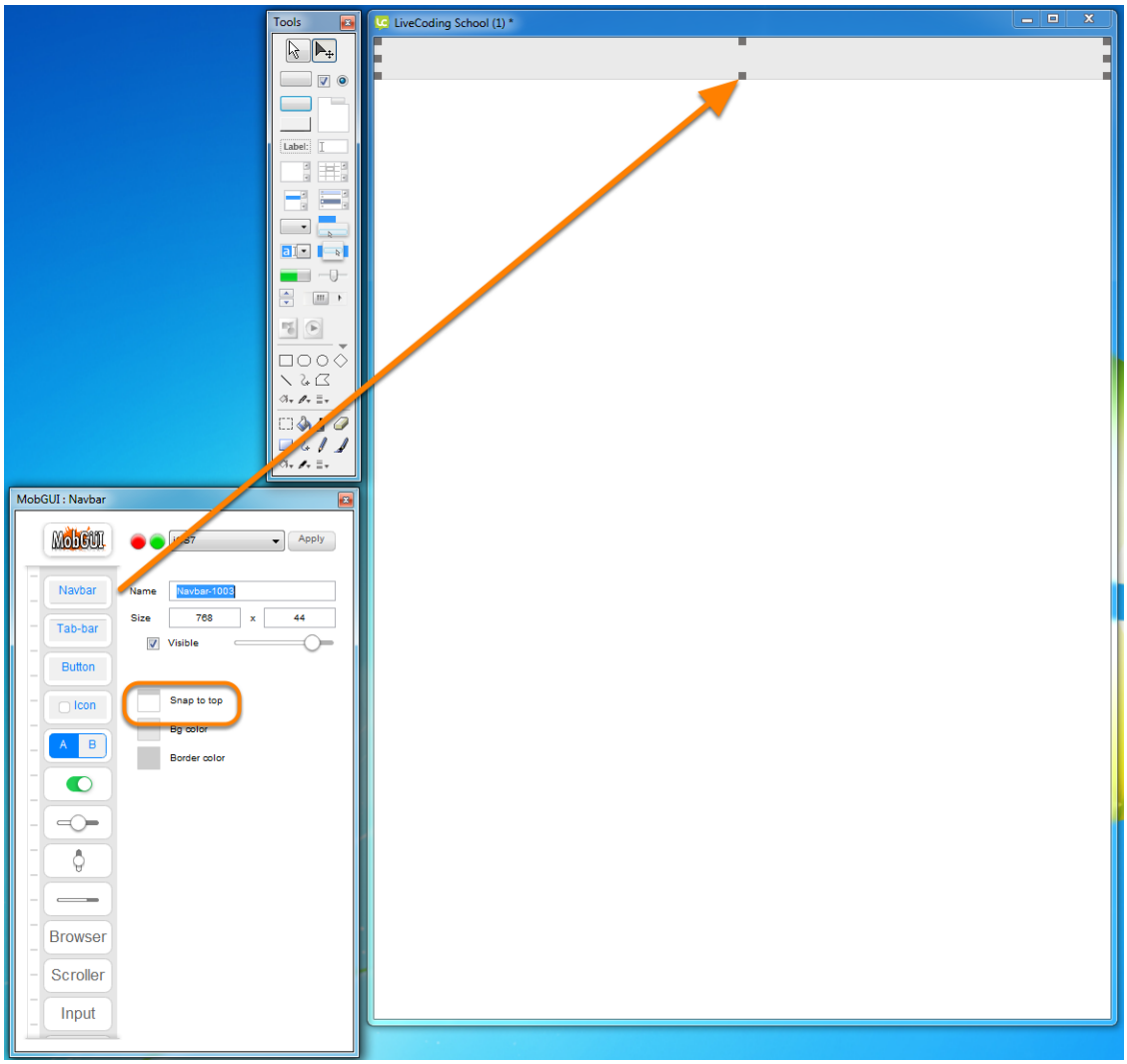
LiveCoding School - Updated MobGUI version

This is a short document and example of the app being built over the course of the LiveCoding School up to the end of Lesson 2. This version uses the current version of MobGUI and will hopefully help explain some of the differences between the versions for those choosing to use the new version.

This document assumes you have created a stack, set it to the correct size, named and saved it.

Adding a Navbar



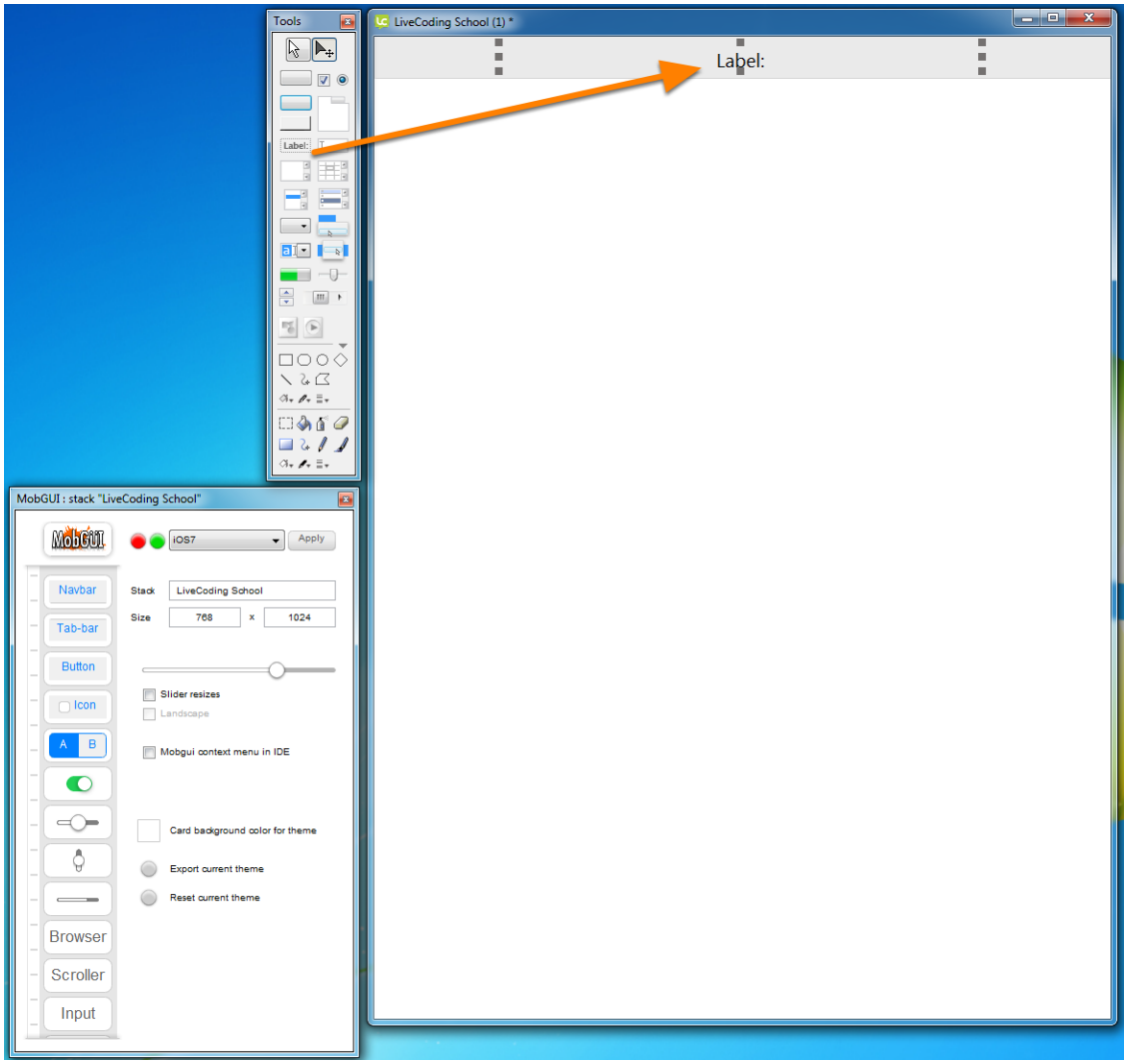
The first control we will be adding is a Navbar. Ensure you are in Edit mode, then drag a Navbar control onto your stack.

With the Navbar control selected click on the "Snap to top" button in the MobGUI palette. This will

resize the Navbar and place it at the top of your stack.

The Navabr doesn't provide any functionality, it is just a background styled for iOS7 or Android.

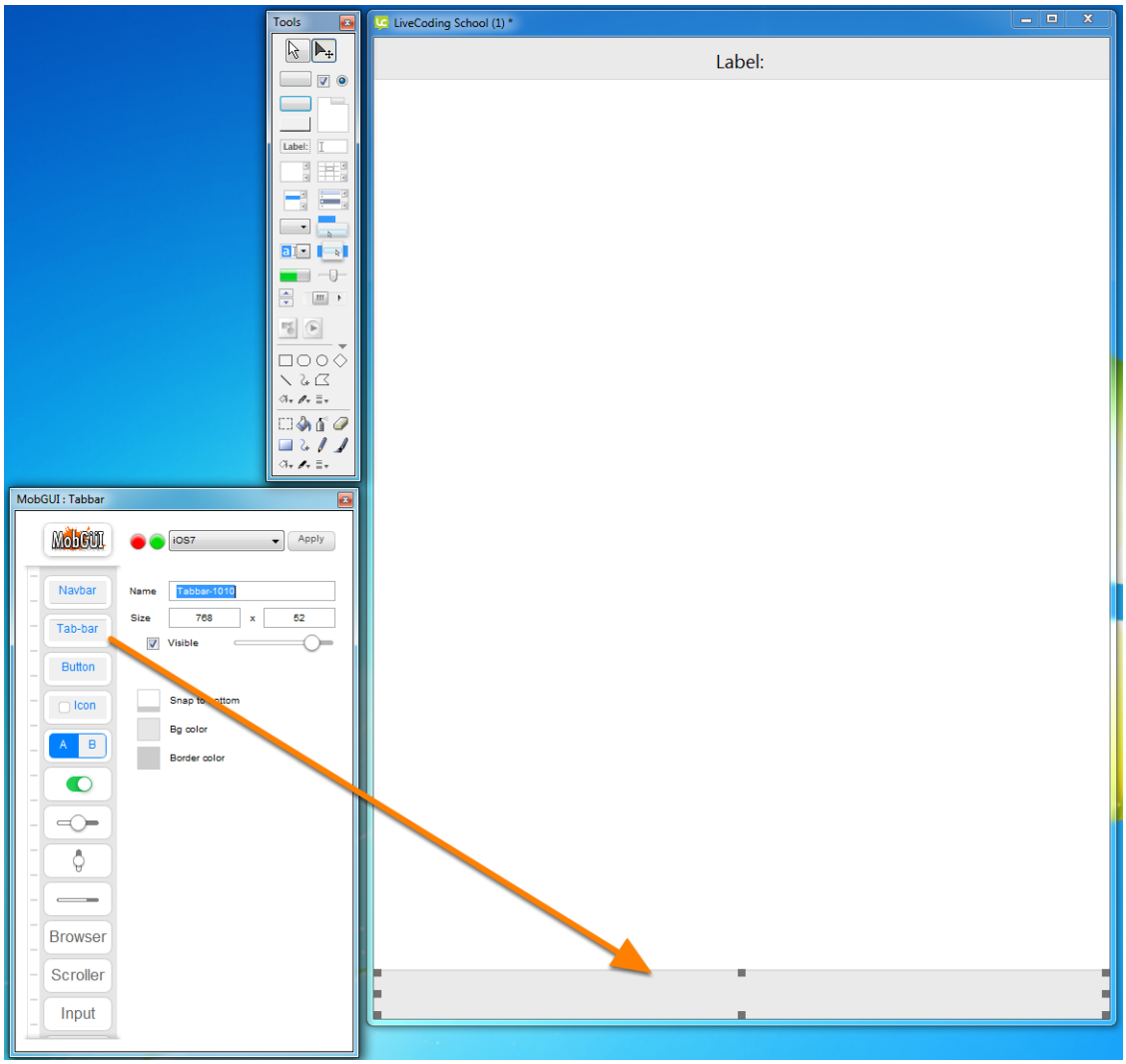
Adding a title field



Next we want to add a field that will display the title of the card the user is on.

Add a label field by dragging it out from the LiveCode Tools Palette. Place it in the center of the Navbar and use the Property Inspector to name the field "title", set the text alignment to center and set the text size and font as you like. Leaving the font empty means it will use the default font, I set the text size to 24.

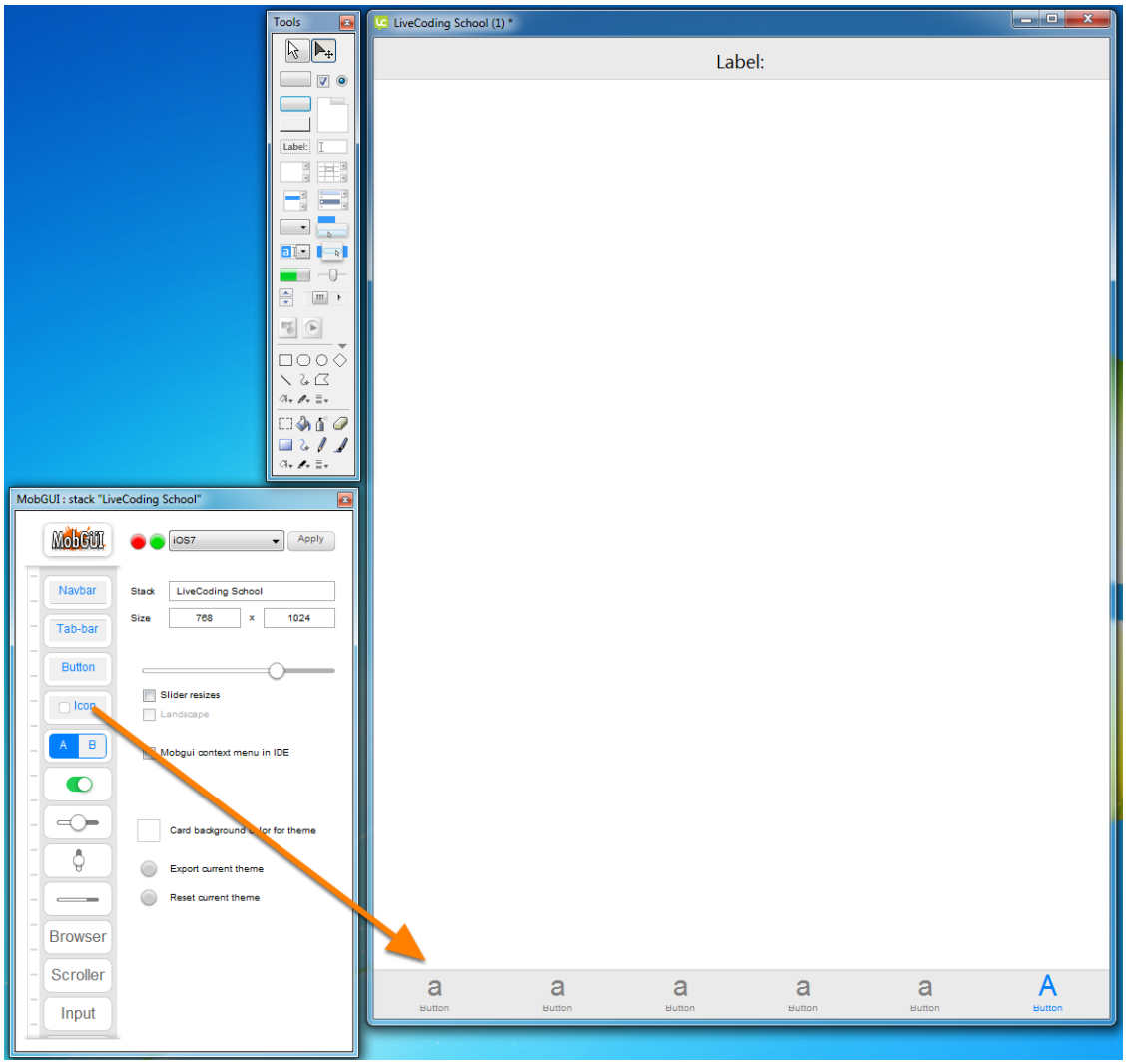
Adding a Tab-bar



The next step is to add a Tab-bar, drag one out from the MobGUI palette, drag it out and click the Snap to bottom button in the MobGUI palette. This will provide a background for the buttons we are going to add.

For more on Navbars and Tab-bars see the [MobGUI docs.]

Creating Icon buttons for navigation



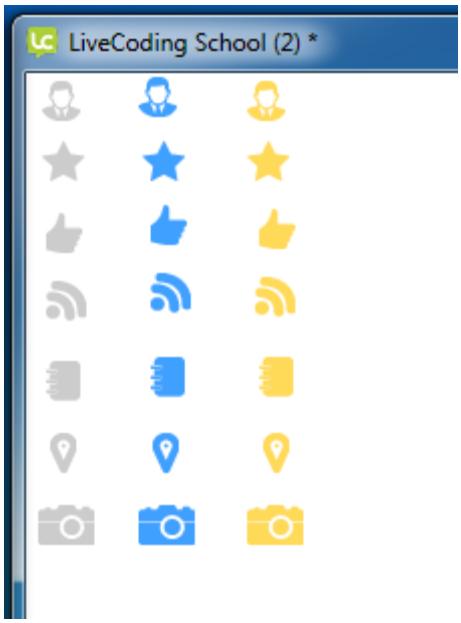
Next we are going to create 6 icon buttons which we will be using for navigating between the cards in our stack.

Drag 6 icon buttons out from the MobGUI palette and arrange them over the Tab-bar at the bottom of the stack.

If you find it easier you can select all 6 buttons and use the Align Objcets pane in the Property Inspector to align and distribute the selected objects in relation to each other.

For more on Icon buttons see the MobGUI docs.

Importing images to use on the Icon buttons



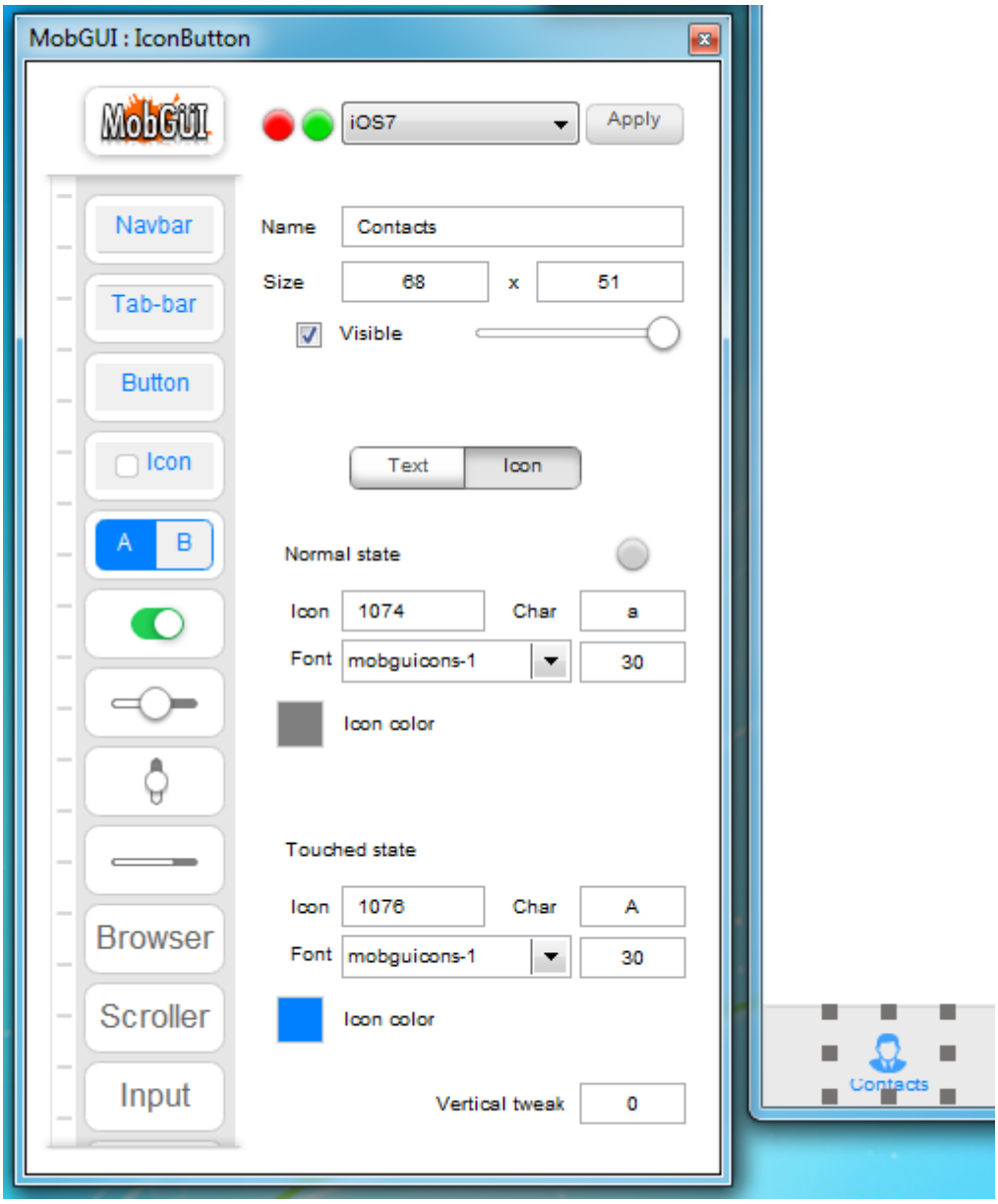
The next step is to import the images we want to use as icons on our buttons into the stack, this allows us to refer to them by a numeric ID. We will store these on a separate card so they are not accessible to the user.

Create a new card by selecting New Card from the Object menu. LiveCode will automatically move to the new card.

To import the icon images select Import as control -> Image File from the File menu and select the images you want to import (these are attache to this post).

Once the images are imported select each one in turn, open up the Property Inspecnotor and take a note of its ID. We will use these IDs to assign images to the Icon buttons we created. Each button will have a normal and a touched state, the grey icons are for the normal state and the colored icons and for the touched state.

Setting up the Icon buttons



Now go back to the first card by selecting Go First from the View menu.

We are going to set up the 6 buttons to represent the cards we will have in our stack. These will be Contacts, Maps, Offers, Camera, Blog and Share.

Select the first button in the MobGUI palette set the Name of the button to "Contacts".

Next click the Text option and set the Text of the button to "Contacts", this text should now appear on the button. You can aslo adjust the font, size etc here.

Now switch to the Icon option and set the Icons for the Normal and Touched states, these Icon

IDs are the ones we took a note of on the previous step.

Repeat this step for the other 5 buttons.

Adding code to the icon buttons

We will be grouping our icon buttons and handling the mouseUp message in the group control, allowing us to implement the code just once. To allow this each of the icon buttons needs to pass the mouseDown and mouseUp messages it receives so they can be handled on a group level.

Select the Contacts button and open the Code editor. Add pass statements into the mouseDown and mouseUp handler, then apply the script. Repeat this for the other 5 buttons.

```
# option buttons usually act as soon as they are touched
on mouseDown
    pass mouseDown
end mouseDown

# normal buttons usually act after a touch
on mouseUp
    pass mouseUp
end mouseUp
```

Creating the navigationButtons group

Next we want to group our 6 buttons together. Select all 6 button and click the "Group" button in the menubar. This will group the buttons together, use the Propety Inspector to set the Name of the group to "Navigation buttons".

Agroup is a set of controls that has been made into a single control. You can select, move, resize, or copy the group, and all the controls in it come with the group. You can show a border around the group (using its **border** property), a label (using the showName property), or scrollbars.

Messages that are received by controls belonging to a group pass to the group before the card. We will use this feature to handle the mouseUp message on the group script, this means we don't have to repeat code on each of the 6 button.

Select the group and open the Code Editor. Add the following code

```
on mouseUp
    local tSelection

    // Store the pressed button in a variable
```

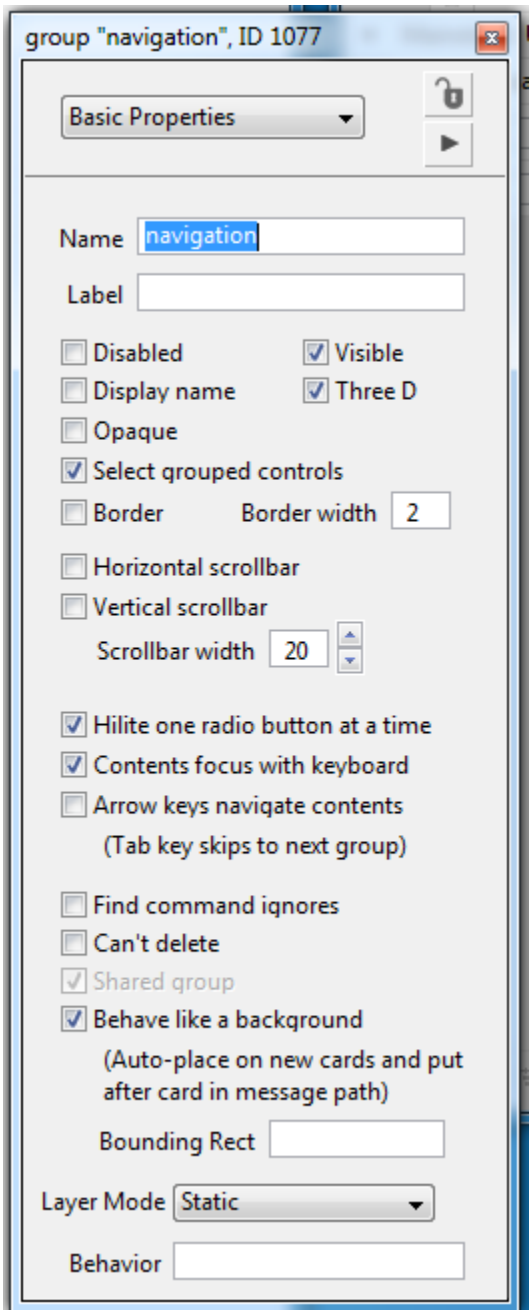
```
    // As the name of our cards match the text of the buttons we can use this to change cards
    put the mgText of the owner of the target into tSelection

    // Set the contents of the "title" field
    put tSelection into field "title"

    // Go to the selected card using a visual effect
    lock screen for visual effect in rect "0,44,768,970"
    go to card tSelection
    unlock screen with visual effect push left fast
end mouseUp
```

Apply the script and save your stack.

Setting up the Naviagtion background



We want one card for each options, before creating out additional card there is one more step we need to take.

All the controls we have created so far will appear on all our cards, this is commong for header, naviagtion etc. So that we don't need to replicate these controls multiple times and implement the same code in multiple places we will group all these controls together and set them as a background. This allows the group to be placed onto multiple cards, any backgrounds will automatically be placed on new cards, you can also add and delete them manually using the place and remove commands.

Select all your controls, the Navbar, "title" file, Tab-bar and "Navigation buttons" group and group them together by clicking the "Group" button in the Menubar. Open the Property Inspector for this group, set the Name to "Navigation" and turn on the "Behave like a background" property.

Adding the additional cards

Now we need to add and name all the additonal cards we will be using in our app. We will start by naming the first card. Open the Card Inspector from the Object menu and set the Name of the card to "Contacts". Now add a new card by selecting "New Card" from the Object menu, LiveCode will automaticall move to the new card and you will see the "Navigation" group is automatically placed on the new card. Name this card "Maps", repeat this for the other 4 cards, ensuring their names match the names of the buttons.

Now switch to Run mode and use the navigation buttons to change cards, you should see the title change but other than that there will be no visible change as only chnages to the central content area are shown.

In the sample stack I have put some colored graphics in the center of the cards so you can see the transitions.