Adding Code to your app

LiveCode uses what is called Event Driven Programming. This means that your app will sit and wait for the user to do something. This is an **event**. LiveCode will tell your app when there is an **event.** You then add code to tell your app what to do when an **event** occurs.

LiveCode lets your app know what happened by sending a **message** to the most relevant control. The message will tell your app what type of event happened and which control it happened to.

For example, when the user clicks on the **tiger** button, a **mouseUp** message is sent to the **tiger** button. You add code to the **tiger** button, telling it what to do when it gets a **mouseUp** message.

There are hundreds of different messages in LiveCode, but you only need to add code for the ones you are interested in sending.

Adding Code to the Stack

In this app, sounds should play when the user presses the buttons. You should have already downloaded the sounds. They will be in the **Resources** folder, in a second folder named **sounds**.

You should have already downloaded the **Resources** folder in the previous step, but **if you haven't, do that now!**

Make sure that the folder named Resources has been moved so that it is in the same folder as your stack.

To play the sounds, LiveCode needs to know where to look for the sound files, so we need to add some code to our app that points to where these files are.

Adding Code to the Stack

Each object in LiveCode can have its own code associated with it. Stacks, cards, buttons, etc, can all have code.

To make sure that our app can find the sound files it needs, we will set the **Default Folder** when the stack is opened. The **Default Folder** tells your app which folder to look in for external files.

Because we want this to happen when the stack is opened, we tell the stack to respond the the **preOpenStack** message, which is sent to the stack when it is first opened.

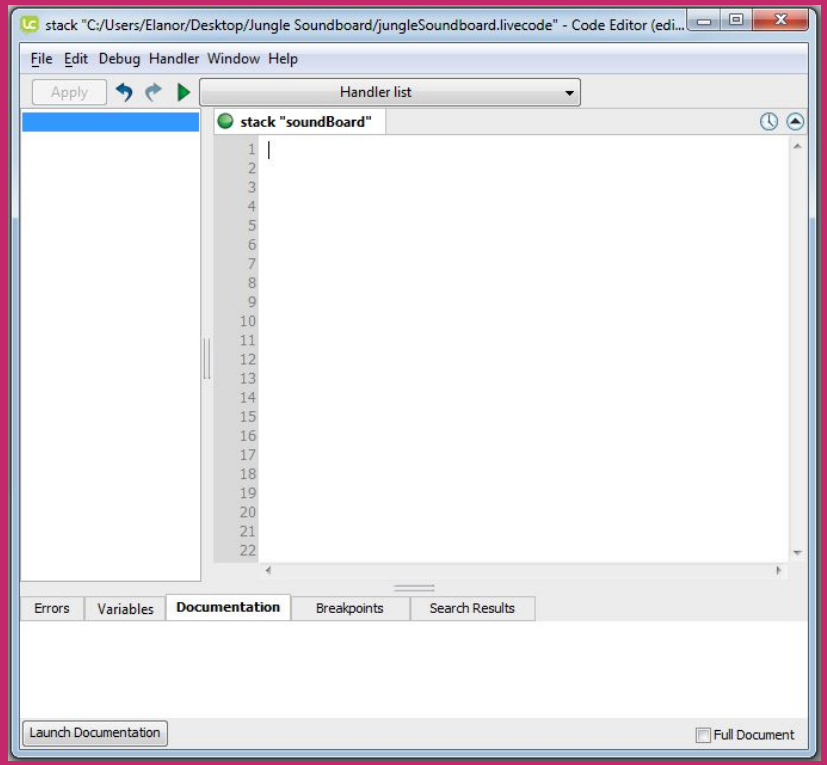
Adding Code to the Stack

To add code to the stack:

1. Open the *Object* menu.
2. Select *Stack Script*.

A new window will open - the Script Editor. This is where you add your code to the app.

You'll see in the tab at the top that you are adding code to the **soundBoard** stack.



Adding Code to the Stack

Type the code below into the Script Editor. Then hit *Apply* (you will find this in the top left corner of the Script Editor).

```
on preOpenStack
    set the itemDelimiter to "/"
    set the defaultFolder to (item 1 to -2 of the filename of me)
end preOpenStack
```

The indicator next to the apply button should go green, showing you there are no mistakes.

If the indicator is red, check your code for mistakes.

If the indicator is yellow, it means you have changed the code and not applied it. Always remember to apply your changes so LiveCode recognises them.

What Does This Code Mean?

1. Just before the stack is opened, execute the code below
2. Set the item delimiter to "/," which allows you to split pieces to text into **items** where the divider between items is "/".
3. Set the default folder - the folder where LiveCode will look for files - to the folder that this stack is saved in
   me - the object the code the belongs to (in this case the stack)
   filename - the full path to the stack file. e.g.
   C:/Users/Elanor/Desktop/Jungle Soundboard/jungleSoundboard.livecode
   so item 1 to -2 will be "C:/Users/Elanor/Desktop/Jungle Soundboard," which is the folder the stack is saved in
4. Stop executing the code

```
1. on preOpenStack
2.    set the itemDelimiter to "/"
3.    set the defaultFolder to (item 1 to -2 of the filename of me)
4. end preOpenStack
```

Execute the preOpenStack message

You want to set the **default folder**. The code that you added will do this when the stack is opened, but our stack is already open.

We can tell LiveCode to send a message manually if we need to.

1. Ensure you are in **Edit mode**
2. Right/ctrl click anywhere on the stack, this will cause a popup menu to show giving you a list of options
3. Choose **Send Stack Message** -> **preOpenStack** from the menu
4. This will cause the **preOpenStack** message to be sent to the stack

Once you have done that, the code that you just added to the stack will have been executed and the default folder set!

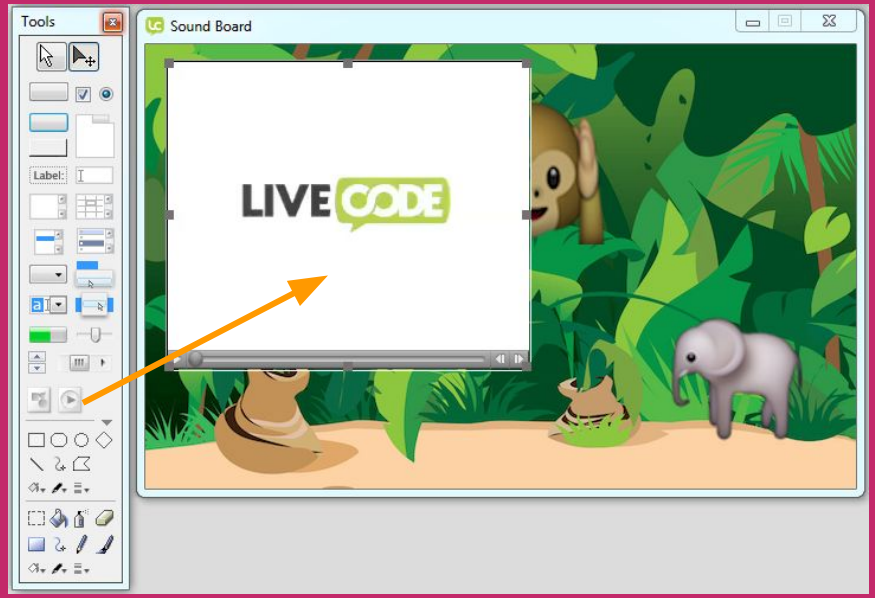
Well done, you are halfway through this step. Swap pains now.

Pains Swap

Add a Player Object

In order to play sounds, you need to add a player object to your app.

Drag out a player object from the tools palette and drop it anywhere on the app.



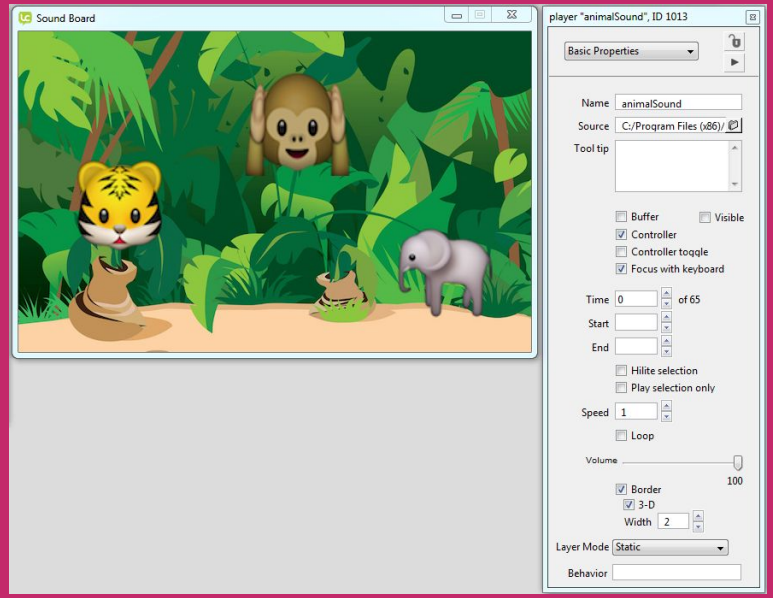
Add a Player Object

You do not want the user to be able to interact with the player, so you want to make it invisible.

Select the player and open the *Property Inspector* by choosing *Object Inspector* from the *Object* menu. Go to the **Basic Properties** pane

- set the **Name** to **animalSound**
- uncheck the box next to **Visible**

The player should now be hidden!



Adding Code to the Tiger Button

Next, we want to add code to the **tiger** button that will cause the tiger sound to be played when the button is clicked.

To do this, we tell the **tiger** button to respond to the **mouseUp** message. This message is sent when the user releases the mouse button.

1. Make sure you are in **Edit mode.**
2. Select the **tiger** button.
3. Select *Object Script* from the *Object* menu.
4. Type the code on the next slide into the **Script Editor**.
5. Hit the *Apply* Button.

The tiger button code

```
1.  on mouseUp
2.    set the filename of player “animalSound” to "Resources/sounds/tiger.mp3"
3.    start player “animalSound”
4.  end mouseUp
```

1. When the user has clicked on this button, execute the following code.
2. Set the source file of the player to the tiger sound. LiveCode will look for this file in the default folder, then in the Resources folder, and then in the sounds folder. This is why it was important to put our sound files in the right place earlier.
3. Start the player - i.e. play the sound file that we have loaded into the player.
4. Stop executing code.

Test the Code

Switch into **Run** mode in the tools palette and click on the **tiger** button.

You should hear the sound of a tiger! Congratulations - you have added functionality to your app! Just one more step and it will be complete.

Now we need to add code to the other two buttons so that we have a fully functioning app!

Well Done

Section Complete

Save Your Work

Pairs Swap