Groups

Just like many design applications such as Adobe Illustrator, LiveCode can select multiple controls and “group” them.

You can then treat the group as a single control and make use of some really useful properties of groups and grouped controls.

Previously we have consolidated our code together into one command to keep the code simple and more stable.

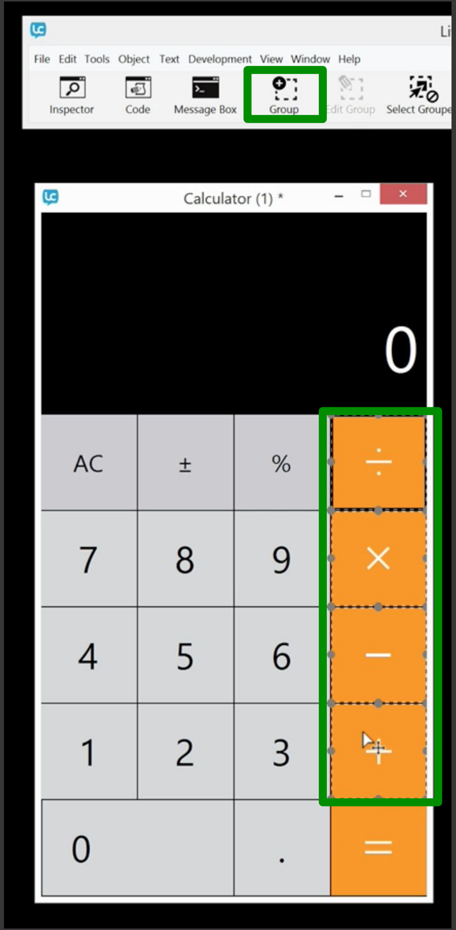
We are going to expand this idea further to include the UI controls of LiveCode. This will show you how your app can benefit from LiveCode’s ability to group controls.

Grouping the Operator Buttons

Today we are going to introduce the concept of control Groups. We are going to group the buttons **divide**, **multiply**, **plus** and **minus** as they have a similar function within the app.

To group the four buttons:

- In edit mode, select the four buttons
- Select the “Group” button in the toolbar
- Select “Object Inspector” from the Object menu - this will open the Property Inspector for the group
- Name the group “operators”



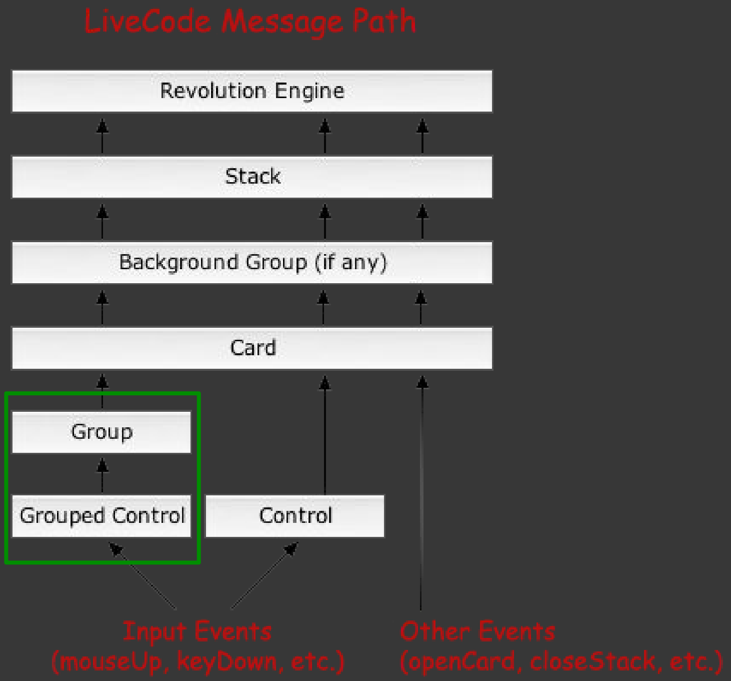
The Group Object

What are the benefits of grouping the buttons?

The buttons are now contained within a single LiveCode object, so we can consolidate our code further!

At the moment, we have four buttons with identical **mouseUp** handlers. As the buttons are now contained within a single object, we can remove the identical handlers from the button scripts and write a single **mouseUp** handler in the group script.

The diagram on the right displays the LiveCode message path, and how groups fit into it.


LiveCode Message Path
Revolution Engine
Stack
Background Group (if any)
Card
Group
Grouped Control
Control
Input Events (mouseUp, keyDown, etc.)
Other Events (openCard, closeStack, etc.)

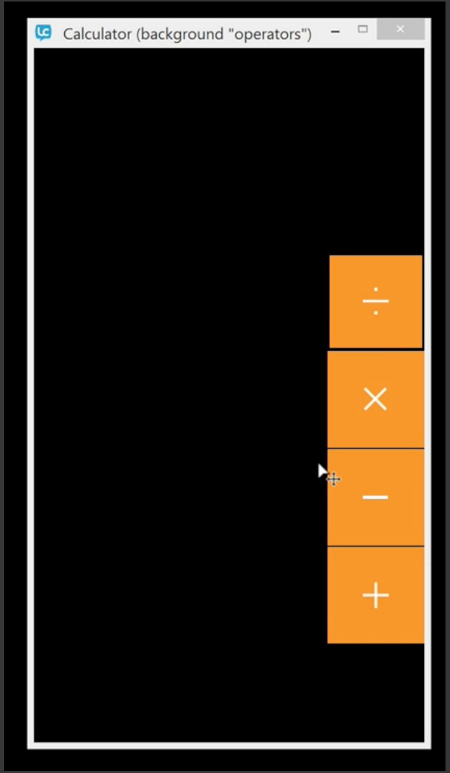
Delete the mouseUp Handlers

Now that we have grouped the buttons, the first step is to delete the script of each button.

To do this:

- select the group in edit mode
- select “Edit Group” from the toolbar
- select each button individually and open the Script Editor
- select all of the code and delete, don’t forget to click the “Apply” button
- select “Edit Group” in the toolbar

You will have noticed that when you are in the edit group mode, only the controls within that group are visible. When you exit the edit group mode the UI returns to normal.



The Group Script

Now we need to add a **mouseUp** handler to the group script:

- Select the group in edit mode
- Select “Object Script” from the object menu
- Add the code on the right

When a control that is in a group is clicked it gets a **mouseUp** message. If the message is not handled in the script of the grouped control, then the message is passed to the group.

When one of the buttons in the **operator** group is clicked, we call the **operatorPressed** command and pass the name of the button that was clicked as a parameter.

The **target** is an in-built LiveCode function. In this case, it returns the button that was clicked in the group that caused the **mouseUp** message to be sent.

```
on mouseUp

 operatorPressed the short name of the target

end mouseUp
```

LiveCode Key Concept

the target - This function returns the object which received the message that started the execution.

the target OR *target()*

Use **the target** function within a message handler to determine which object originally received the message.

Suppose a card script contains a **mouseUp** handler. If the user clicks a button, a **mouseUp** message is sent to the button. If the button's script does not contain a **mouseUp** handler, the message is passed to the group (or the card, if the button does not belong to a group) that the button belongs to, and is handled by the group's **mouseUp** handler.

The Group Script

LIVECODE Calculator

Notice the slight change in the code when moving the script of the buttons into the group.

In the script of the buttons we had:

```
operatorPressed the short name of me
```

versus

```
operatorPressed the short name of the target
```

in the group script.

The reason for this change is that we want to pass the short name of the button that was clicked as a parameter to the **operatorPressed** command.

In the button script, we can use the **me** keyword but if we used this in the group script, then we would be passing the short name of the *group* and not the *button* to the **operatorPressed** command. We have to use the **target** function so that we can retrieve the button that was clicked.