LiveCode – a basic introduction

by Julian Morgan

Warning: working with LiveCode can be very addictive!

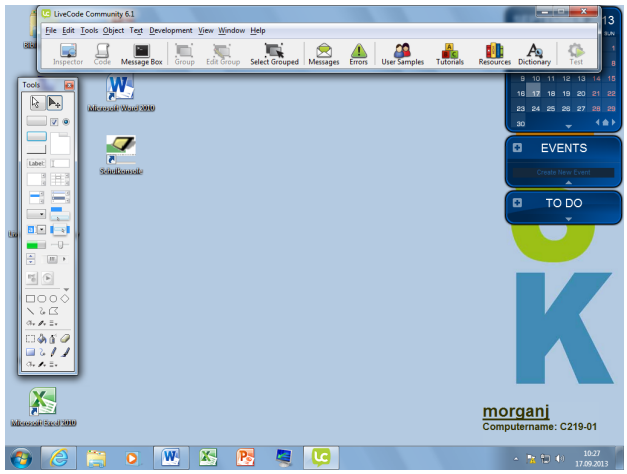
When I first bought a Macintosh computer in 1990, it was for one basic reason: HyperCard. This authoring system was revolutionary at the time and quite wonderful to use. But gradually, Apple lost interest in it and it was dumped at the beginning of the new Millennium. Luckily, a company in Scotland called Runrev brought it back to life, renamed it (twice) and it has now become an important development platform for software which will run on PC's, Macs, Linux, iOS and Android devices. And it has very recently become free of charge in the so-called Community version.

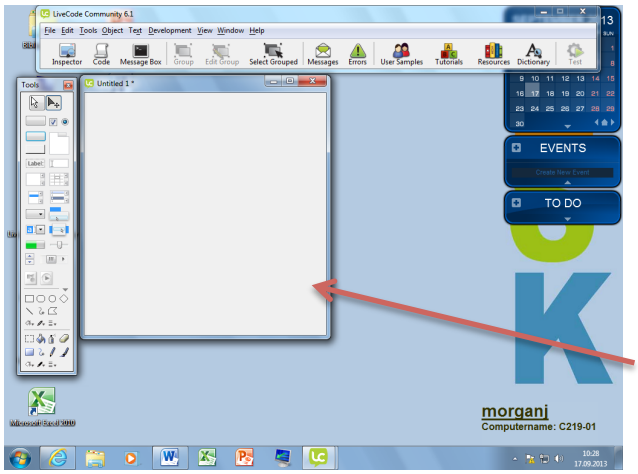
Just a few words of caution, however. If you want to develop professionally on all of those platforms, it will cost you a lot of money and among other things, you'll have to buy a Mac to do all the iOS programming. So don't get too excited. What you will be able to do will be to write software for use in your own classroom and school environment, which can also be shared with your students. That in itself was not possible before, so be glad.

LiveCode is based on a card metaphor (Powerpoint more or less took this metaphor over when it first came out) and is very easy to understand. In this workshop, we shall be trying to create a few cards in LiveCode, which will do different jobs. We won't get far in a few hours, but hey! Let's give it a go anyway.

Task 1: Understanding Cards and basic tools

Start off by launching LiveCode on your computer and then close the Start Center screen. What you will see will be this – or something similar.



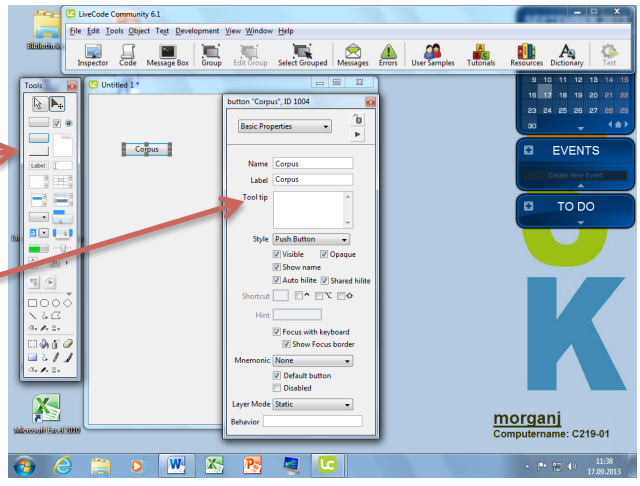


Now Choose New Mainstack from the file menu, and what you will see will be something like this. Try moving the edges of the new card around and you will see that it is resizable.

Now, from the Tools palette, take a Default button and drag it across onto the card you have made.

Double click on the button and you will get a dialog, where you can give the button a name. Try typing "Corpus" into the box twice, once for the Name and once for the Label.

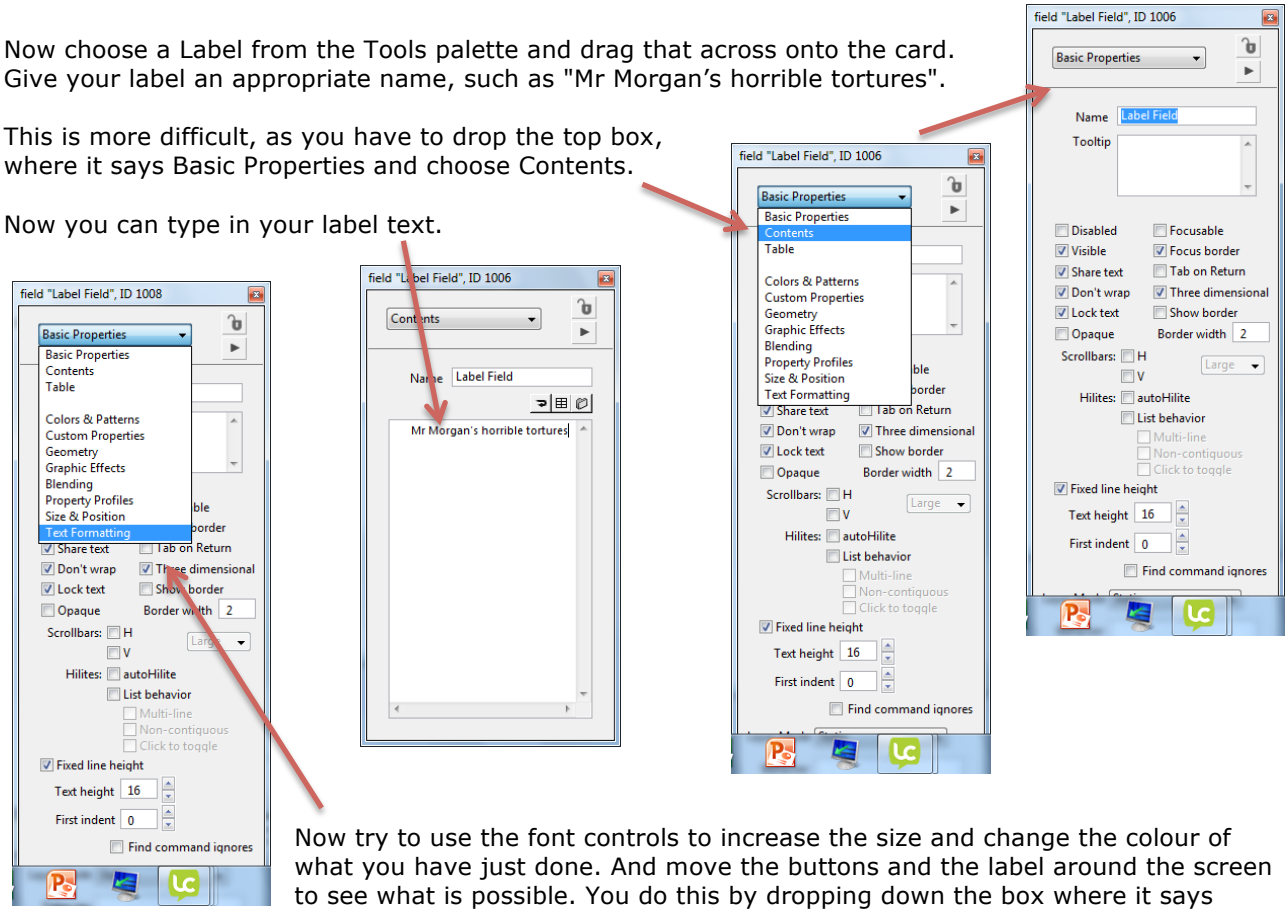
Now repeat the process by making another button, which you will label "Vocab Test" or similar.



Now choose a Label from the Tools palette and drag that across onto the card. Give your label an appropriate name, such as "Mr Morgan's horrible tortures".

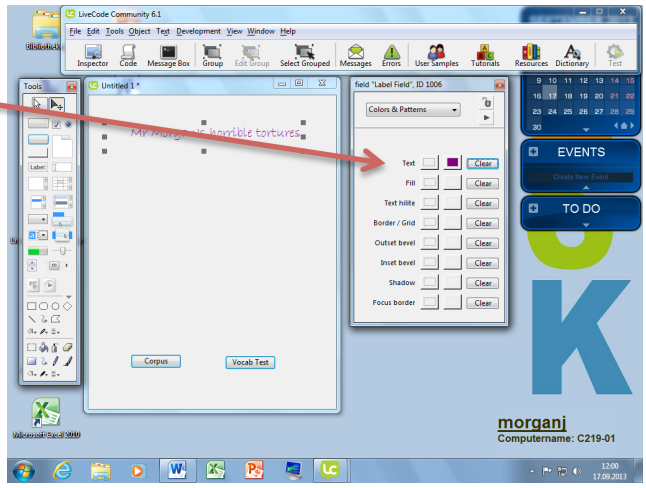
This is more difficult, as you have to drop the top box, where it says Basic Properties and choose Contents.

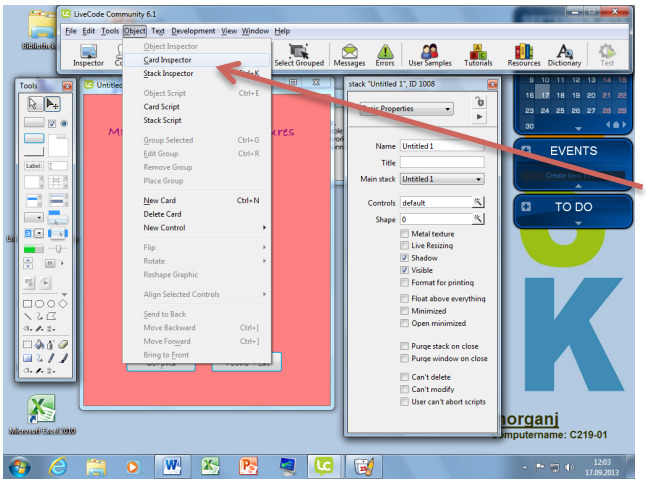
Now you can type in your label text.



Now try to use the font controls to increase the size and change the colour of what you have just done. And move the buttons and the label around the screen to see what is possible. You do this by dropping down the box where it says Contents and selecting Text Formatting.

If you want to change the colour you have to go back to the same drop-down and choose Colors and Patterns. One of the best things about this system is that all the cards are blank when you start and what it looks like when you finish is entirely your creation.



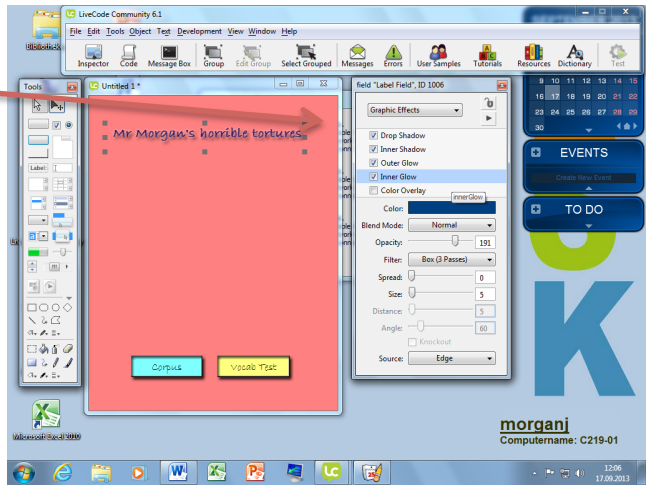


Now go to the Object Menu and choose Card Inspector. See if you can change the background colour of the card you have made. Try resizing the buttons and making new font sizes, colours and background fill colours for each one.

Now select each button with a single click, and where it says Style, make each one a rounded rectangle. You are beginning to get the idea that the property inspector is basically the same for cards, buttons and indeed all elements in LiveCode.

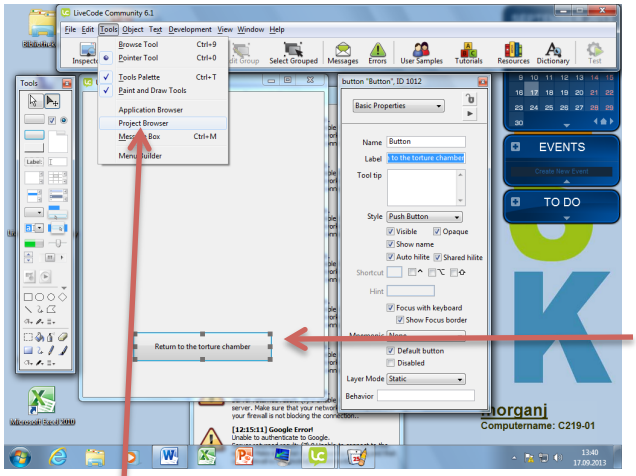
Try selecting each of your three objects in turn (label and two buttons) and experiment with giving them some Graphic Effects.

In the screenshot, I gave the buttons a Drop Shadow and an Inner Shadow, whereas the Label had slightly different things done to it. Watch out with giving too many effects! They look great but they slow down iPads to an impossible crawl.



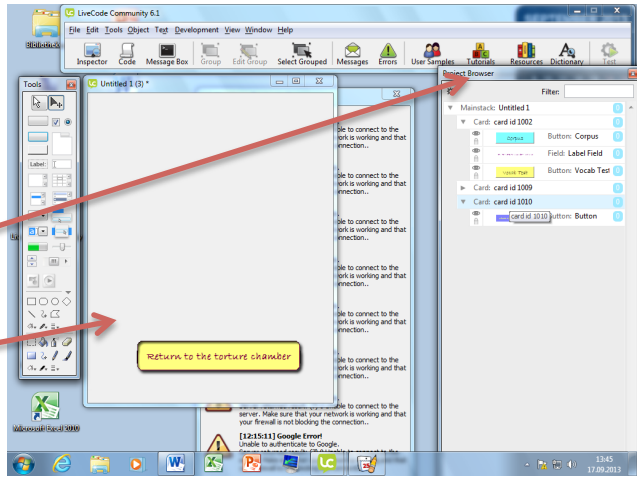
Now go to the Object menu and make two new cards in turn. Each card will be white and blank.

On one of the two new cards, you should create a button which says "Return to the torture chamber".



In case you can't find the card or how to get there, use the Project Browser pane. If this is not visible, you can view it by selecting Project Browser, in the Tools menu, as shown.

Just double click on where it gives the card id, eg card id 1010, seen here, and you will find yourself going there.



In case you have forgotten already, you should place the words "Return to the torture chamber" in the Label box of the new button, once you create it. You can add some colours and effects if you like.

Now you are ready for some programming. (We are about to use a system where all new texts to be typed are shown in purple: if it isn't purple, leave it alone.)

Click once on the button you just made and then click where it says Code, on the main top bar. A window will now open up, for the code to be typed.

Type:

```
on mouseup
```

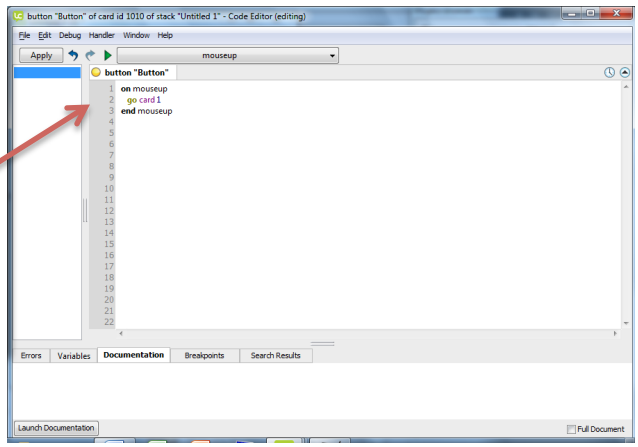
You will then see

```
end mouseup
```

appear by magic below.
Click between the lines and type:

```
go card 1
```

It should look like this.



Now click on the button which says Apply, close the window, and see what happens when you click the button. You will probably have to click on the top left button (the Run button) in the Tools Palette, to switch from Edit Mode to Run mode, or this may not work.

You will need to get used to doing this! And whenever you want to write code, you will have to switch this control back to Edit Mode again.

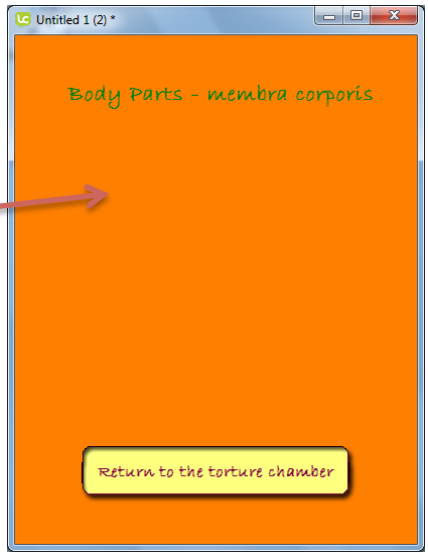
Now insert some code to make LiveCode go to Card 2, when the Corpus button is clicked, and to go to card 3 when the Vocab Test button is clicked. Make sure you have tested all buttons.

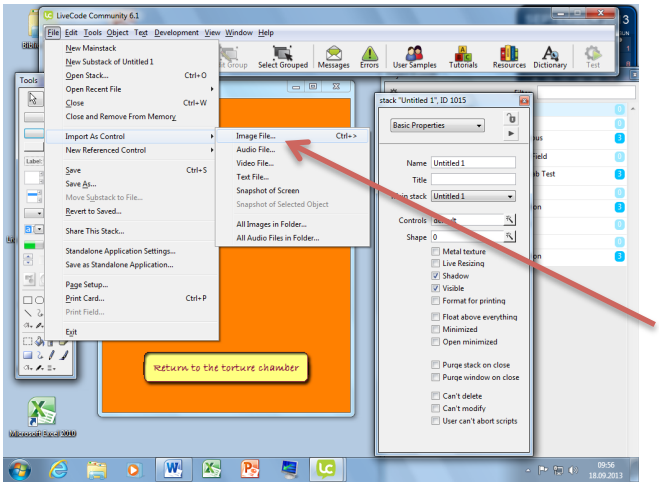
You will need to copy and paste the button you just made (it is probably on Card 3) so this appears on both of the white cards.

Congratulations. You have just built the bare bones of a working LiveCode stack. It's a start.

Task 2: the Corpus Card

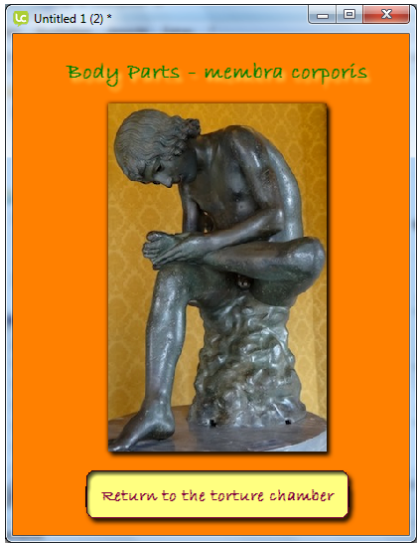
Now make sure you are on the Corpus card. This will become a simple quiz, based around an imported graphic and a simple blank button. It's probably a good idea to insert a Label, with an introduction to the card on it. If you do this, make sure you choose a font, a colour, perhaps a style and place it where it looks good. Also, select the Card Inspector from the Object menu and choose a background colour for the card. Play around with fonts and effects and see if you can make something which looks like this.



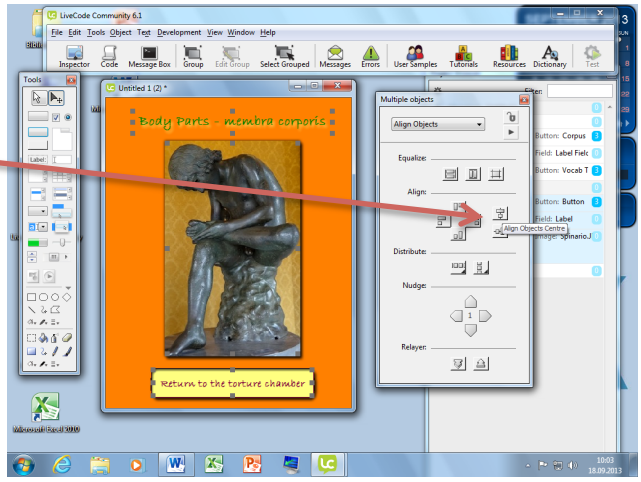


Now go to the File Menu, and select Import As Control. You are going to insert an image, which can form the basis for a quiz. You will need to choose the first option here, Image File.

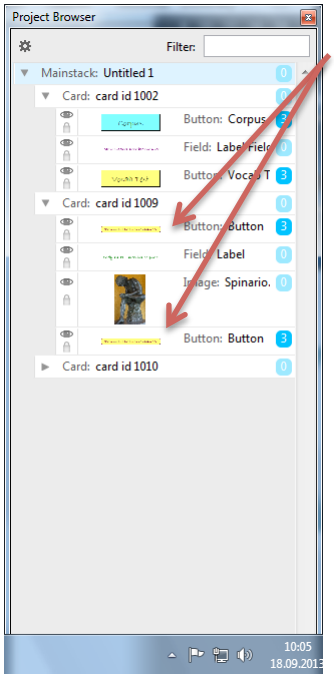
Now find the image called Spinario (or use one of your own) and insert it. After this, give it some nice graphic effects, such as a Drop Shadow and an Inner Shadow. And move everything round a bit, so the card looks lovely.



If you select the text at the top, the image and the button at the bottom, by first clicking on one of them, then holding down the SHIFT key as you click on the others, you will see the Align menu appear. From here you can align the horizontal centres by clicking on the button, as seen here.



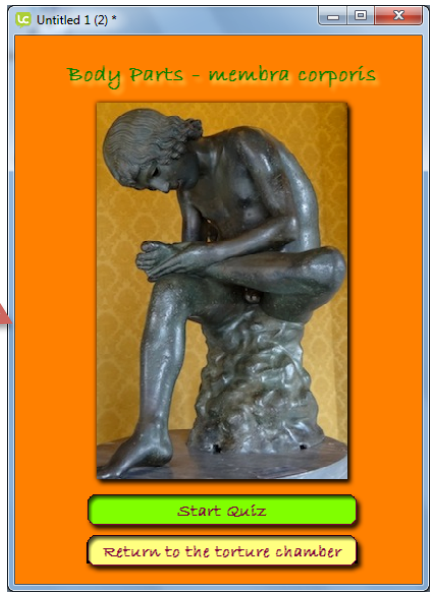
Now you need a new button, to start the quiz. Why not just copy and paste the one you already have? To do this, click away from the aligned group you just selected, and now click the button once again. Now do the normal Copy/Paste routine. You should get a new button – but in case you can't see it, that will be because it's on top of the one you just copied.

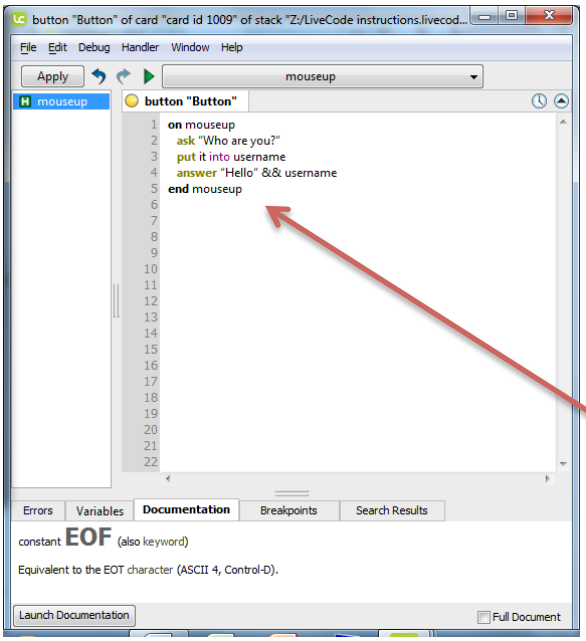


The Project Browser will show you if it's there.

NB You may have to do some resizing and moving things around now.

Now move the new button above the first one, change its colour to something else, and change its name to Start Quiz. All of this should now look like this:





Now click once on the Green, Quiz button, and you will see the script. This is where the fun starts.

Where it says go card 1, delete the text and insert instead:

```
ask "Who are you?"
put it into username
answer "Hello" && username
```

Now click on Apply, close the window and run the script. If it works, well done. If it doesn't, try again.

A word about variables

The thing you just made, which is called username, is a variable. There are different types of these – ones which run in individual scripts, and ones which run in any script. The best ones to users are the ones which run in any script, called global variables. Now modify what you just typed by adding in one new line and then saving the script as you did before.

```
global username
ask "Who are you?"
put it into username
answer "Hello" && username
```

A word about ampersands

The ampersand was apparently invented by Cicero's secretary, Tiro. It is used to concatenate (join together) two words or variables. In the example above, there are two ampersands - &&. The first joins the word Hello to the variable username, but we also need the second one, to create a space between them. Otherwise it will say HelloFred, rather than Hello Fred.

You can perhaps use this knowledge to tidy things up a bit now. Change the line

```
answer "Hello" && username
```

to

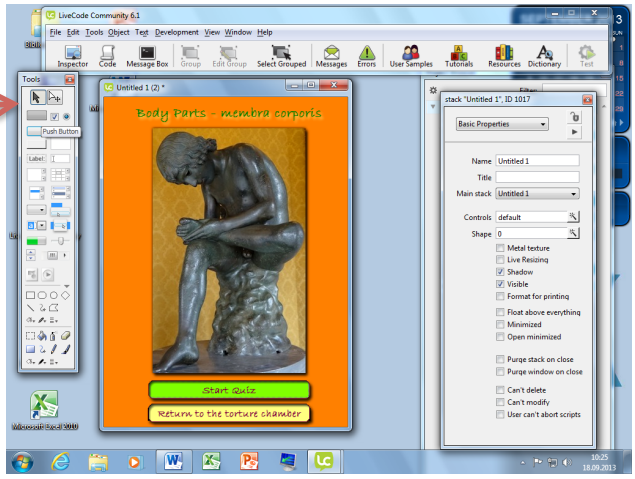
```
answer "Hello" && username & "."
```

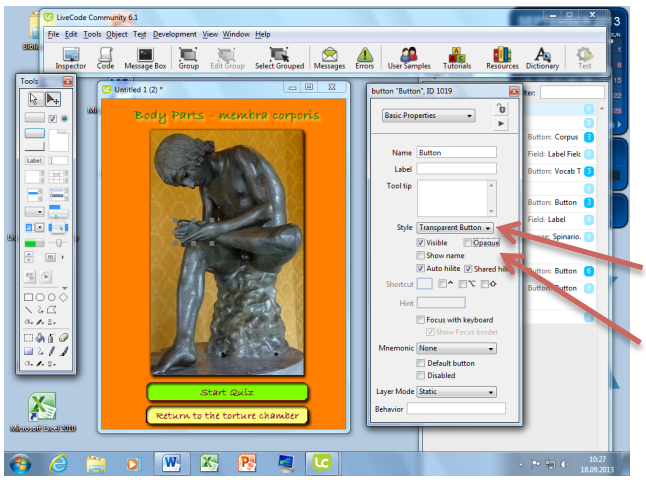
Now run the script by clicking on the button and you'll see it looks a lot neater. Things like this make a huge difference, of course.

Now let's have some fun and change to Latin. Change the script, so it reads like this:

```
on mouseup
   global username
   ask "quis es?"
   put it into username
   answer "salve" && username & ". imprime manum."
end mouseup
```

Now you need to create new button, which we will make invisible and place over the hand in the picture. Just drag a normal Push Button from the Tools palette so it covers the hand, and make it generally the right size.





When you have done that, double click it. Now in the palette, where you can see the basic properties of the button, select as a Style, Transparent Button.

Also, deselect Show Name and Opaque.

Now make sure you have selected the Edit button in the Tools palette and click your new, transparent button. Click on Code in the bar above. Now type in a script, which should read:

```
on mouseup
   global username
   answer "bene fecisti," && username & "."
end mouseup
```

Now test it. This is great but very limited, because the button can actually be pressed at any time at all. So first of all, give the new button a name, Hand. You can do this in the Project Browser or the Object Inspector.

Now change the script of the Start Quiz button to read this:

```
on mouseup
  show button "Hand"
  global username
  ask "quis es?"
  put it into username
  answer "salve" && username & ". imprime manum."
end mouseup
```

But now, you have to be sure the button is hidden again after it has been clicked, so amend the script of the button "Hand" to read

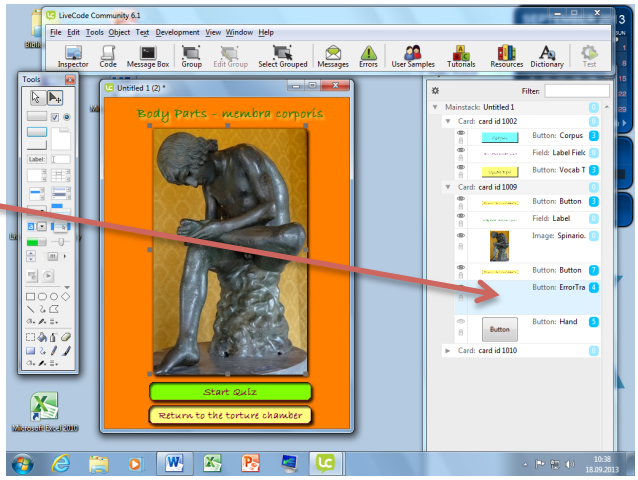
```
on mouseup
  global username
  answer "bene fecisti," && username & "."
  hide button "Hand"
end mouseup
```

Now see if you can do one more thing. Create a new transparent button, called "ErrorTrap". Give it the same properties as your button "Hand" and make sure it extends across the whole of the picture. Give it a script which says this:

```
on mouseup
  global username
  answer "erravisti," && username & "."
end mouseup
```

Now try to run the program. You should find it will only give the error message. This is because your new button is bigger than the one for the hand, and it is on top of it.

To correct this, click on the button ErrorTrap in the Project Browser and move it above the button "Hand". Now try again and it should work.



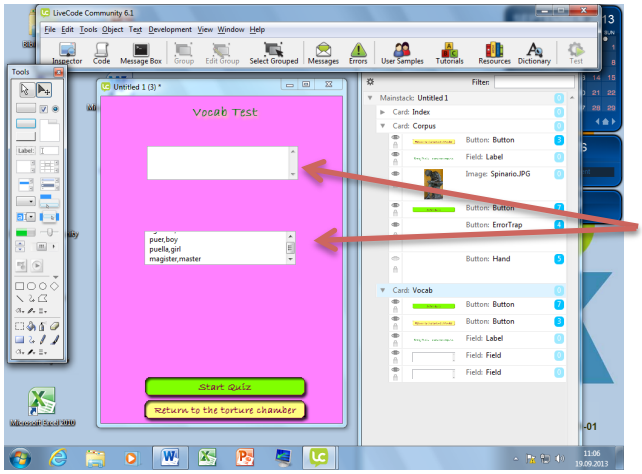
Of course, you are just touching the surface here, but even so, you may begin to wonder whether you couldn't build something useful out of such easy programming techniques.

Task 2: the Vocab Quiz

This will once again be kept short, so here goes. First of all, go back to the main, first card, and tidy it all up a bit, in view of the new size of your stack (a stack is the name given to a collection of cards). It may be a good idea to use the Project Browser to give names to the cards now. Call the first card Index, by clicking in the Browser Window where the new name should go. Now you can see which card is being worked on as you go.

Do the same for the two other cards.

It is generally a good idea just to give the card a simple, one-word name, so I am calling my Vocab Test card simply *Vocab* here.



Now copy the two buttons from your *Corpus* card and paste them into the *Vocab* card. Oh, and the label field too, so when the user comes here, everything will look like one unity. Rename all your various bits.

Now drag two scrolling fields onto the card, as seen. A field is the name given to a text container. Name one of them Answers, and the other one Qdata.

Now make some simple questions and type them into the Qdata field, eg

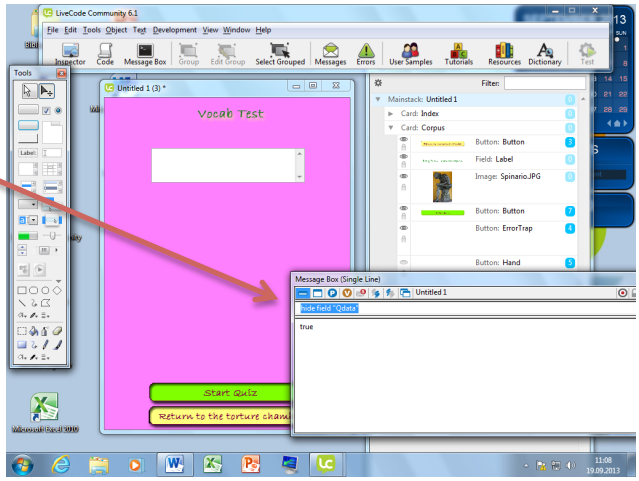
ego,I
agricola,farmer
puer,boy
puella,girl
magister,master

NB – at this stage, don't make more questions than this. Don't try to run before you can walk.

Now click on where it says Message Box, on the top bar, next to Code. Type into the message box:

```
Hide field "Qdata".
```

Now press return.



The question data field should now be invisible.

By the way, in any of these scripts, you can type btn instead of button, cd instead of card, fld instead of field.

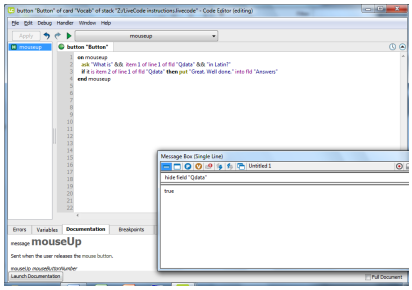
Now click on the Edit button in the main Tools palette, click on Code in the top bar and start on your script for the green button.

Let's start by typing a script like this (as before, I am using purple to indicate the bits which are new each time):

```
on mouseup
 ask "What is" && item 1 of line 1 of fld "Qdata" && "in English?"
end mouseup
```

Now run the script and see if it works. Yes? Then continue to adapt it, like this.

```
on mouseup
 ask "What is" &&  item 1 of line 1 of fld "Qdata" && "in Latin?"
 if it is item 2 of line 1 of fld "Qdata" then put "Great. Well done." into fld "Answers"
end mouseup
```



Still working? OK, try this.

```
on mouseup
 ask "What is" &&  item 1 of line 1 of fld "Qdata" && "in Latin?"
 if it is item 2 of line 1 of fld "Qdata" then put "Great. Well done." into fld "Answers"
 wait for 2 seconds
 put "" into fld "Answers"
end mouseup
```

Of course, you can start thinking about how to adapt this model as you go. But what we need to do is to create a system of repeating the questions, for as often as there are questions to ask. We do this by creating a variable x, which will correspond to the number of lines which exist in the field "Qdata". We do this by changing the script, as follows.

```
on mouseup
   repeat with x = 1 to the number of lines of fld "Qdata"
     ask "What is" &&  item 1 of line x of fld "Qdata" && "in Latin?"
     if it is item 2 of line x of fld "Qdata" then put "Great. Well done." into fld
     "Answers"*
     wait for 2 seconds
     put "" into fld "Answers"
   end repeat
end mouseup
```

* *Where you see texts spill over onto a second line in these instructions, don't let this happen in your code.*

Now try running your script. If it's working, great. If not, try to correct it.

We now need responses for incorrect answers. Try something like this:

```
on mouseup
   repeat with x = 1 to the number of lines of fld "Qdata"
     ask "What is" &&  item 1 of line x of fld "Qdata" && "in Latin?"
     if it is item 2 of line x of fld "Qdata" then put "Great. Well done." into fld
     "Answers"
     if it is not item 2 of line x of fld "Qdata" then put "Oh dear. I think you meant"
     && item 2 of line x of fld "Qdata" & "." into fld "Answers"
     wait for 2 seconds
     put "" into fld "Answers"
   end repeat
end mouseup
```

This should work, but it's not very elegant. Also, since we want to put in a score handler, it's going to get pretty horrible unless we tidy things up. Try to change things so they look like this:

```
on mouseup
   repeat with x = 1 to the number of lines of fld "Qdata"
      ask "What is" &&  item 1 of line x of fld "Qdata" && "in Latin?"
      if it is item 2 of line x of fld "Qdata" then
         put "Great. Well done." into fld "Answers"
      else
         put "Oh dear. I think you meant" && item 2 of line x of fld "Qdata" & "." into
         fld "Answers"
      end if
      wait for 2 seconds
      put "" into fld "Answers"
   end repeat
end mouseup
```

Now, let's insert a scoring system. Change the script so it looks like this:

```
on mouseup
   put "" into fld "Answers"
   put 0 into userscore
   repeat with x = 1 to the number of lines of fld "Qdata"
      ask "What is" && item 1 of line x of fld "Qdata" && "in Latin?"
      if it is item 2 of line x of fld "Qdata" then
         put "Great. Well done." into fld "Answers"
         add 1 to userscore
      else
         put "Oh dear. I think you meant" && item 2 of line x of fld "Qdata" & "." into
         fld "Answers"
      end if
      wait for 2 seconds
      put "Your score is" && userscore into fld "Answers"
   end repeat
end mouseup
```

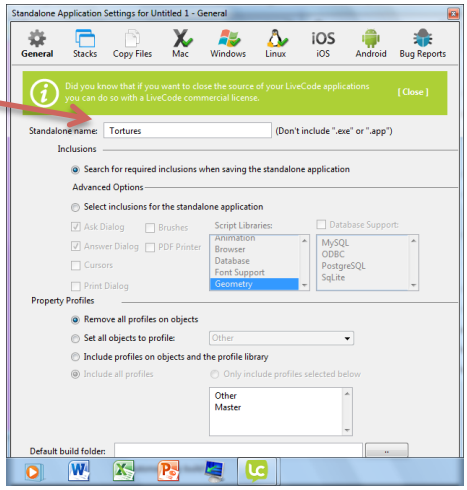
The indents should appear by themselves, if your script is compiling correctly. All you have to do for this is press the TAB key on your computer. The indents make it easier to write code, once you understand how they work.

OK, enough for now. If you want to explore further, then obviously you could experiment with inserting the username into all the question/answer responses. You could tidy up all the fonts and appearance of the thing. A lot of this is stuff you should know how to do from the first part of the workshop. You could also add a randomising system, so the questions get put into a different order every time, which is more of a challenge. If you are getting that far, you are probably hooked.

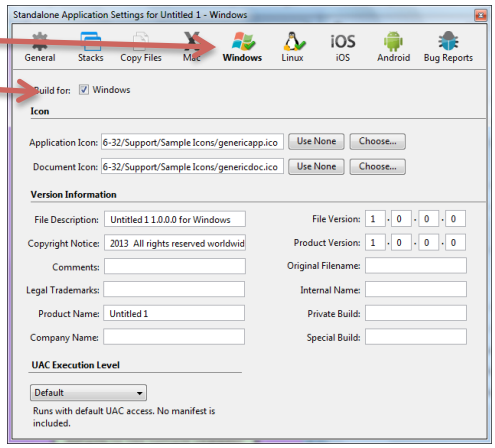
Exporting your work so it runs as a standalone program on a PC

Go to the File Menu and choose Standalone Application Settings…

In the box, give your work a name, eg Tortures



Now click where it says Windows
and make sure there is a green tick in the Build for box.



Go to the other boxes and make sure there is no tick there (or OK, you can build for Mac and Linux as well, if you want. But not iOS or Android – this has to be done separately).

Now go back to the File Menu and select Save as Standalone Application…

Choose a location for your files and hit the OK button!

Congratulations. You have just successfully programmed a usable resource in LiveCode, which will run on any PC anywhere.

Comments or questions? Email julian@imperiumlatin.com

Julian Morgan has been writing software since 1985 on a variety of platforms. He is Head of Classics at the European School of Karlsruhe in Germany and his materials are well-known to many teachers of Classics. (See www.imperiumlatin.com and www.j-progs.com)