(https://livecode.com) Platform (https://livecode.com/products/livecode-platform/) Resources (https://livecode.com/resources/)

Pricing (https://livecode.com/products/livecode-platform/pricing/) Services (https://livecode.com/services/) Blog (https://livecode.com/blog/)

login (https://livecode.com/login/)

Dictionary (https://livecode.com/resources/api/) Guides (https://livecode.com/resources/guides/) Lessons (https://lessons.livecode.com/) Courses (https://livecode.com/products/learn/)

Story Maker

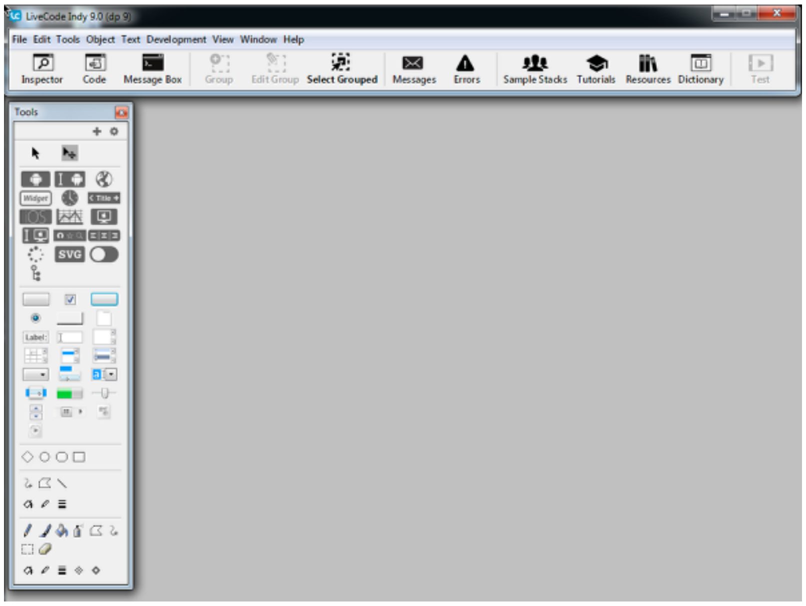
This app allows the user to create a small story by choosing the character's name and description. The app demonstrates building a GUI, working with text, asking the user for input, animation and sounds.

owl_resources.zip

You can download all the resources for this lesson above, or from this link:

https://tinyurl.com/y9rdot3u (https://tinyurl.com/y9rdot3u)

Opening LiveCode



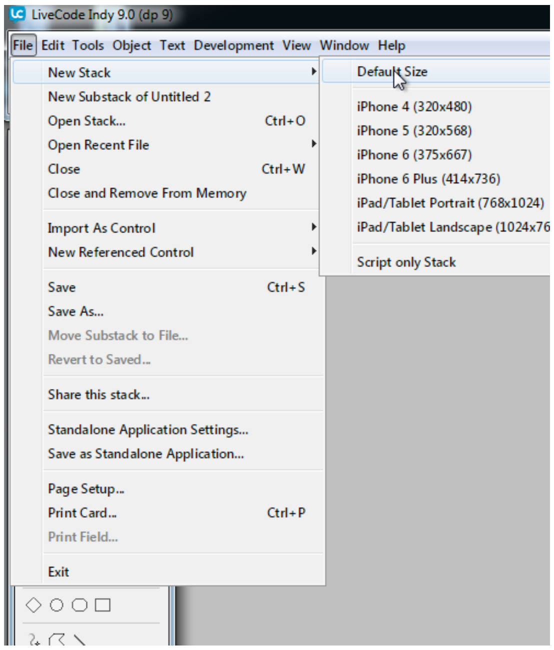
When you open LiveCode you will see the Integrated Development Environment (IDE).

LiveCode uses drag and drop UI creation, an English-like coding language and a compile free iterative development method.

Stacks and Cards

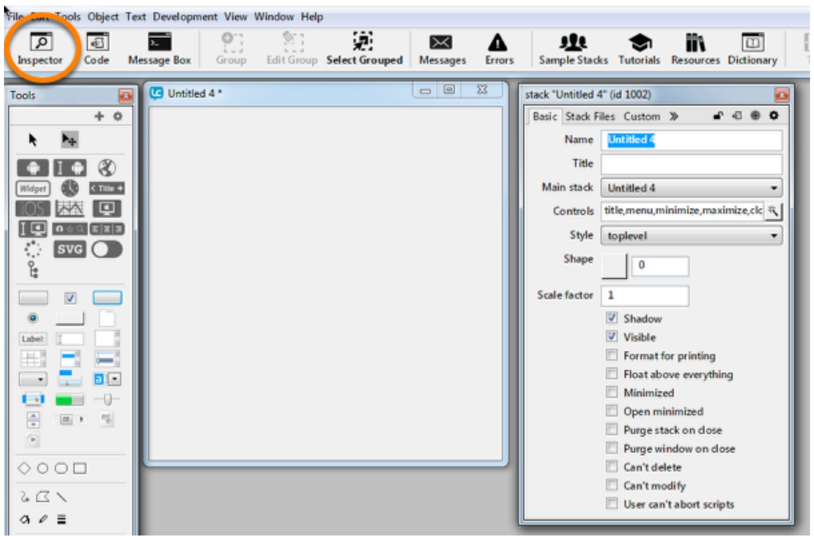
LiveCode works with a stack and card metaphor. Each app you create is a stack, each stack consists of one or more cards. Each card can have a different appearance and you move between the cards to change what the user sees.

Creating a new stack



To start a new app you first create a new stack. Select **New Stack -> default size** from the File menu.

The Property Inspector

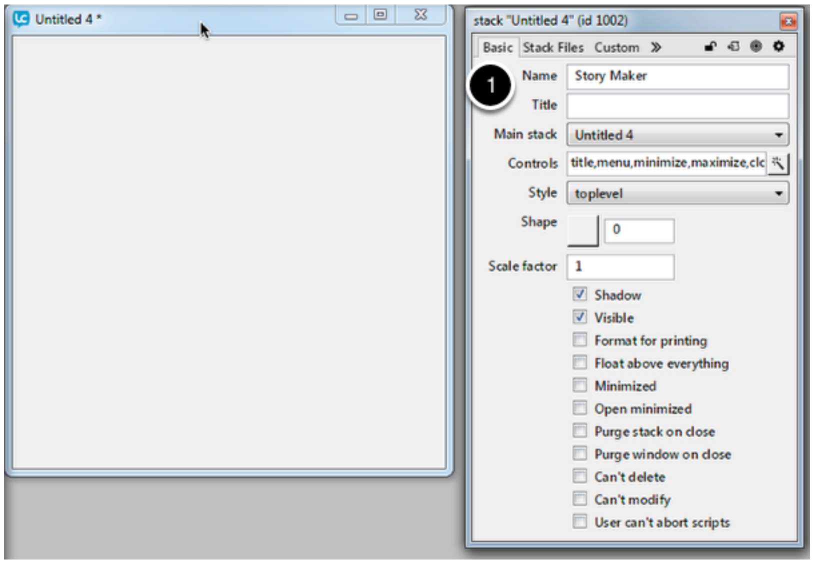


Name: Untitled 4
Title:
Main stack: Untitled 4
Controls: title,menu,minimize,maximize,clc
Style: toplevel
Shape: 0
Scale factor: 1
[✓] Shadow
[✓] Visible
[ ] Format for printing
[ ] Float above everything
[ ] Minimized
[ ] Open minimized
[ ] Purge stack on close
[ ] Purge window on close
[ ] Can't delete
[ ] Can't modify
[ ] User can't abort scripts

Each object in LiveCode has a set of properties that control how it looks and behaves. All of an objects properties can be edited in the Property Inspector or in script.

The Property Inspector can be opened via the Object menu or by clicking the Inspector button in the Menubar.

Naming the stack



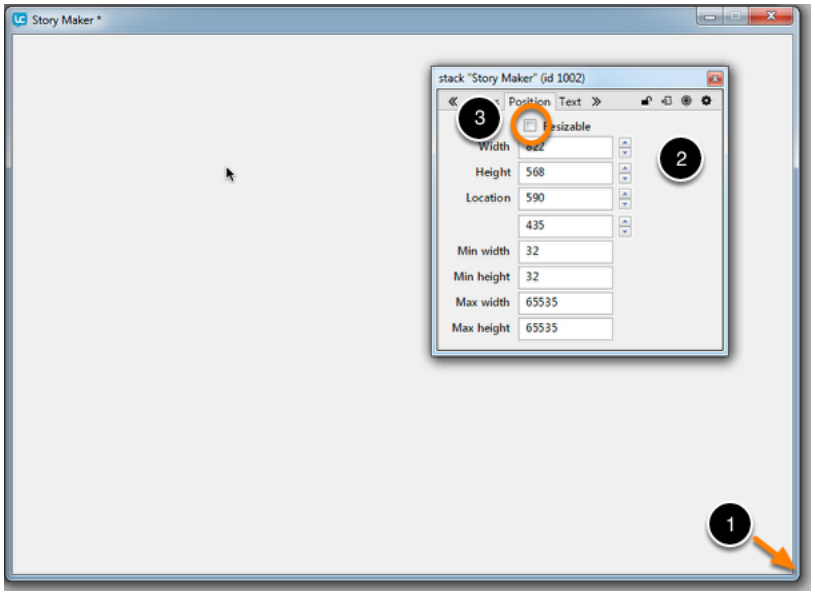
Name	Story Maker
Title	
Main stack	Untitled 4
Controls	title,menu,minimize,maximize,clc
Style	toplevel
Shape	0
Scale factor	1

[✓] Shadow
[✓] Visible
[ ] Format for printing
[ ] Float above everything
[ ] Minimized
[ ] Open minimized
[ ] Purge stack on close
[ ] Purge window on close
[ ] Can't delete
[ ] Can't modify
[ ] User can't abort scripts

Next you want to name the stack. This is done by opening the Property Inspector for the stack and setting the name property to "Story Maker" (1), or whatever you choose to name your app.

Open the Property Inspector for the Stack by choosing Stack Inspector from the Object menu.

Resizing the stack



[ ] Resizable

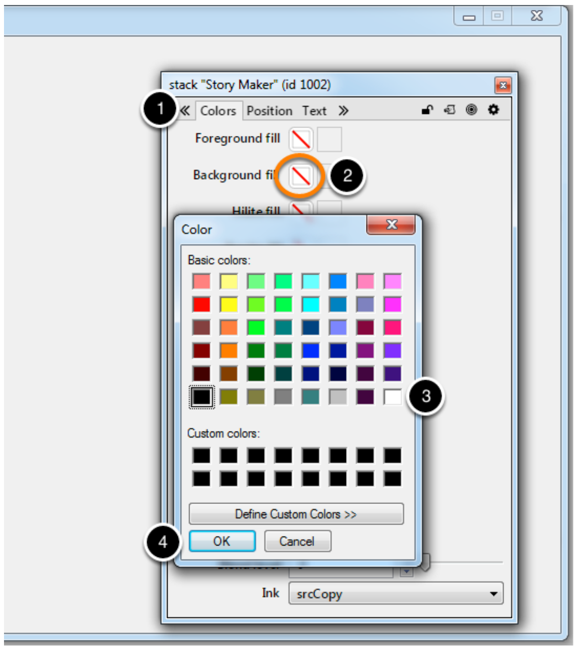
Width	822
Height	568
Location	590
	435
Min width	32
Min height	32
Max width	65535
Max height	65535

Next resize your stack to your preferred size. You can do this by dragging the corner(1) or by choosing the Size and Position pane in the Property Inspector(2) and setting the width and height.

If you are planning on deploying to a mobile device it is a good idea to set the stack to the correct size at this stage. The size for an iPad is 1024 x 768. You can choose this instead of the default size, when you create the stack.

You can also turn off the Resizable property so that you don't change the stack size by accident (3).

Setting the stack background colour



Next we want to set the background colour of the stack to white, all cards will inherit this background colour.

- change the Property Inspector to show **Colors** (1)

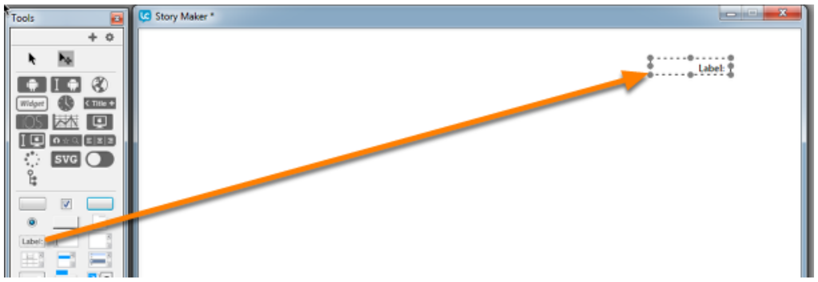
- click the left hand box next to **Background**(the other is for a background pattern) (2)

- choose the colour you want from the colour selector (3)

- click **OK** (4)

The background colour of the stack should have changed to the selected colour.

Adding a field to the stack - 1

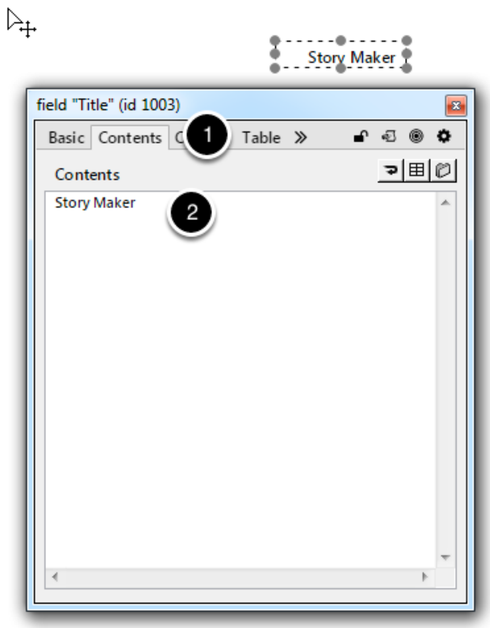


Next we want to add a field to our stack to display the title "Story Maker".

Drag a label field from the tools palette onto the stack. This creates a field object that you can then work with.

A label field is not editable by the user so is a good choice for a title.

Adding a field to the stack - 2



Now we want to add some content to the field

- Select the field.

- Open the Property Inspector for the field by clicking the Inspector button on the Menubar or double clicking on the field.

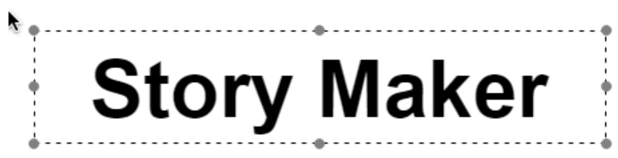
- Set the **name** property of the field to "title" - note the **name** is how we refer to the field in code, what is displayed in it is a separate property.

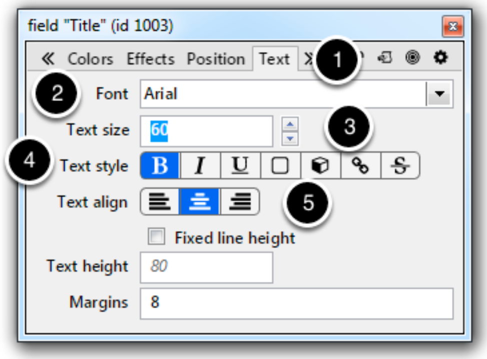
- Change the Property Inspector to show **Contents -** this is where we enter what is to be displayed in the field(1)

- Type "Story Maker" into the text box(2)

Your field should now display the title "Story Maker"

Adding a field to the stack - 3


Story Maker


field "Title" (id 1003)

Font: Arial
Text size: 60
Text style: B I U
Text align
Fixed line height
Text height: 80
Margins: 8

Now we want to make our stack look more like a title

- Change the Property Inspector to show **Text** (1)

- Choose a font to use (2)

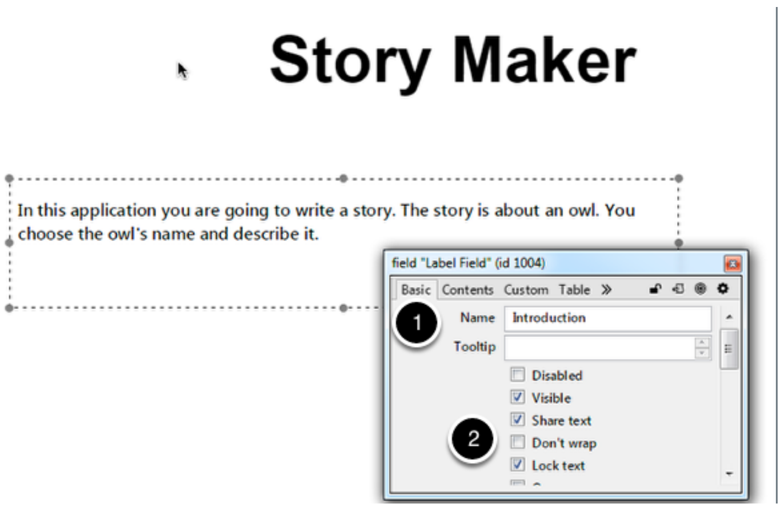
- Set the text size to something big, maybe 60 (3)

- Set the text style to bold (4)

- Set the alignment to centre (5)

You will probably notice that the field is now too small to show all the text so grab one of the handles and drag to make it bigger. Then place your title where you want it to be.

Adding a description field to the stack


Story Maker

In this application you are going to write a story. The story is about an owl. You choose the owl's name and describe it.

field "Label Field" (id 1004)

Name: Introduction
Tooltip:
Disabled
☑ Visible
☑ Share text
Don't wrap
☑ Lock text

Next we want to add a description of what the app will do. Repeat the steps above to create a second field that contains the text

"In this application you are going to write a story. The story is about an owl. You choose the owl's name and describe it."

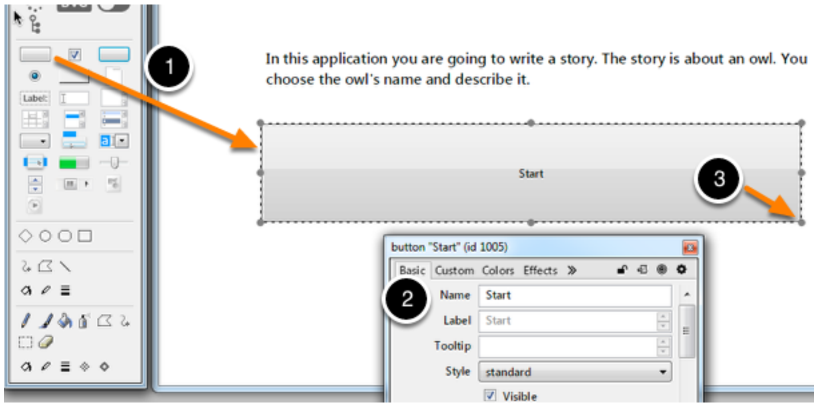
- Give the field a descriptive name (1)

- Left align the text on the text settings pane

- You will probably want to make the text size of this field smaller than the title

- By default label fields don't wrap the text so turn off the **dontWrap** property in the Basic Properties pane to allow the text to go onto more than one line. (2)

Adding a button to the stack - 1



In this application you are going to write a story. The story is about an owl. You choose the owl's name and describe it.

button "Start" (id 1005)

Basic Custom Colors Effects

Name: Start

Label: Start

Tooltip:

Style: standard

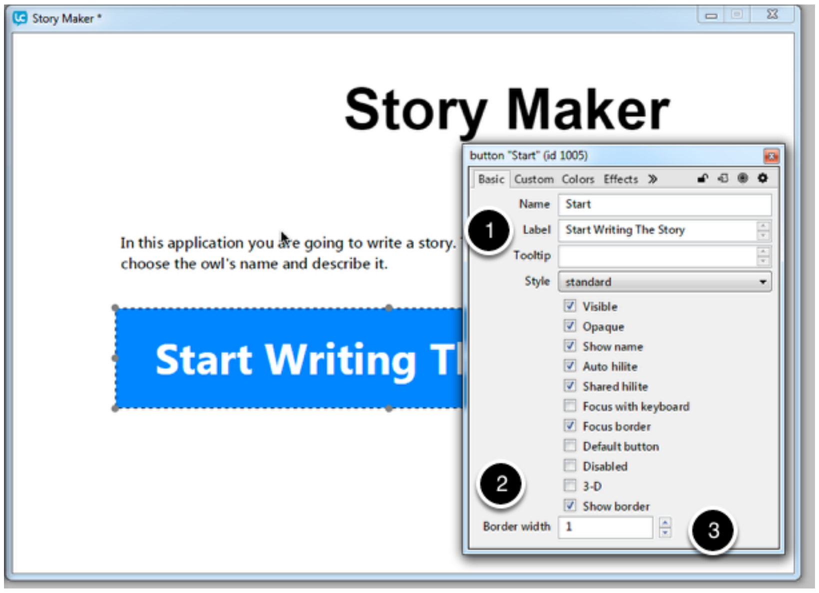
✓ Visible

Next we want to add a button that the user clicks to start writing the story.

Drag a button out from the Tools Palette onto the stack (1).

As before use the Property Inspector to name the button (2) and then resize the button as you wish (3).

Adding a button to the stack 2



Story Maker

In this application you are going to write a story.
choose the owl's name and describe it.

Start Writing Th

button "Start" (id 1005)

Basic Custom Colors Effects

Name: Start

Label: Start Writing The Story

Tooltip:

Style: standard

✓ Visible
✓ Opaque
✓ Show name
✓ Auto hilite
✓ Shared hilite
☐ Focus with keyboard
✓ Focus border
☐ Default button
☐ Disabled
☐ 3-D
✓ Show border

Border width: 1

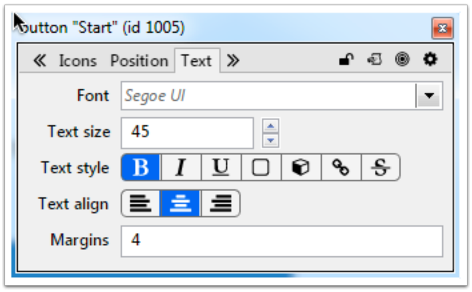
Similarly to the field we will use the Property Inspector to change how the button looks.

- A button can have a **name** which is used to refer to it in script, and a **label** which is the text that is displayed on the button. Set the label of the button to "Start writing the story" (1).

- Turn off **Three D** (2)

- set the **Border width** to 1 (3)

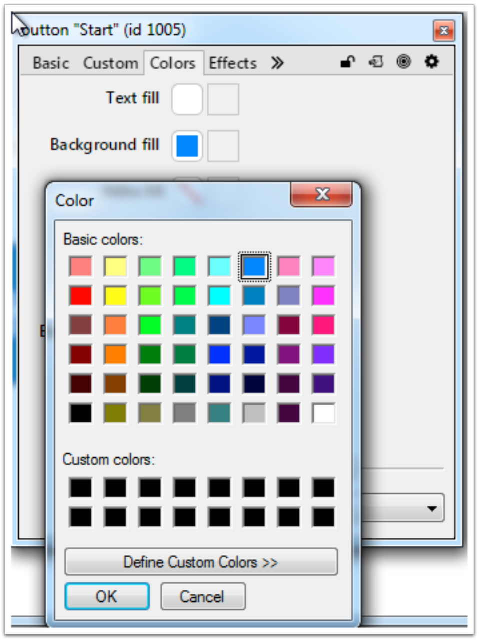
- Change the Property Inspector to the **Text** pane



Font: Segoe UI
Text size: 45
Text style: B
Text align: centre
Margins: 4

- Make the **Size** larger

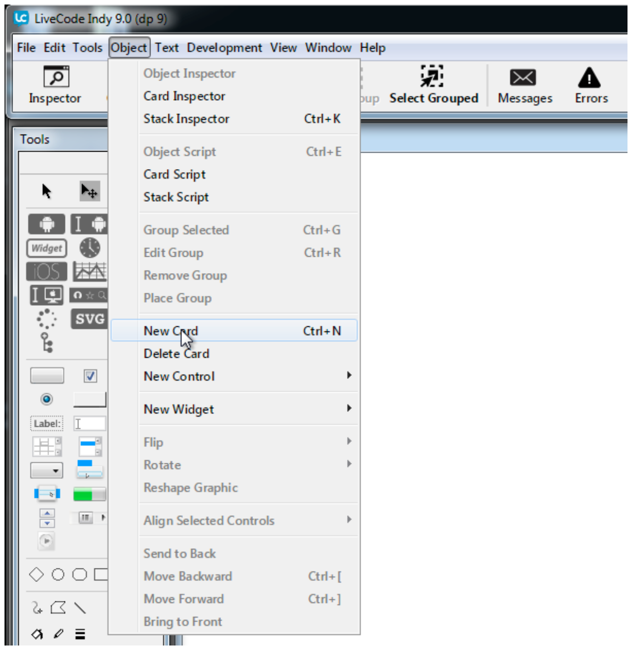
- Set the **Style** to bold



- Change the Property Inspector to the **Colors** pane

- Choose a **Fill Color** and **Text Color** for your button.

Adding a second card

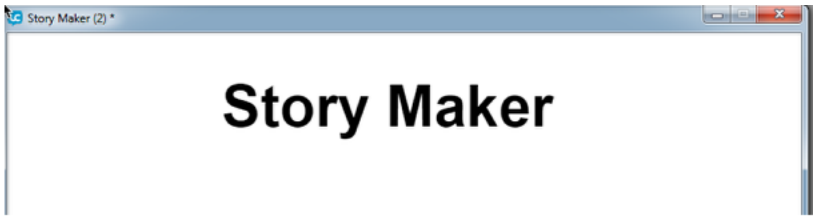


Now that we have set up the introduction card we need a second card that will show the story.

Add a second card to the stack by choosing **New Card** from the Object menu.

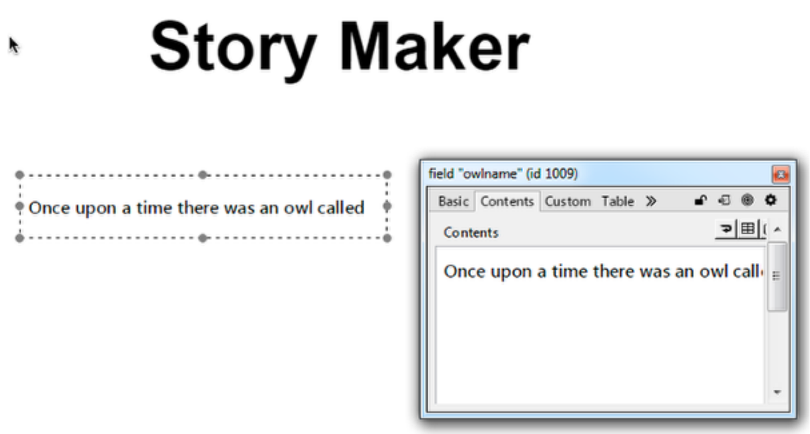
LiveCode will automatically switch to the new card so when everything goes blank that's fine.

Setting up the second card



This card will have a title, the same as the first card, so follow those steps again to create the title field.

Setting up the story fields - 1


Story Maker
Once upon a time there was an owl called
field "owlname" (id 1009)
Basic
Contents
Custom
Table
Contents
Once upon a time there was an owl call

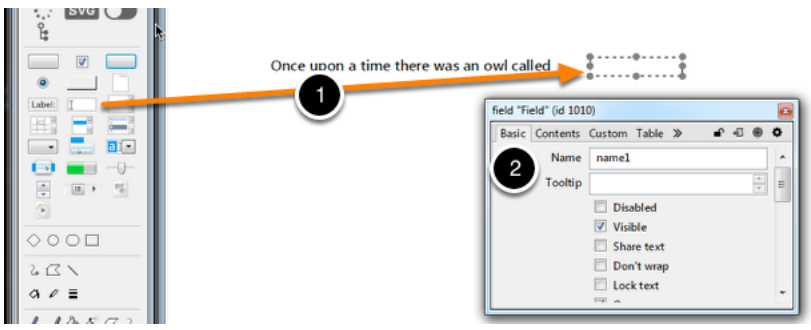
Next we need to add some field to tell the story and fill in the owl's name and description.

- Add a label field to tell the first part of the story

- The contents of this field should be something like "Once upon a time there was an owl called"

- Set up the display properties of the field using the Property Inspector as before.

Setting up the story fields - 2



Next we want to add a field that will display the name chosen by the user.

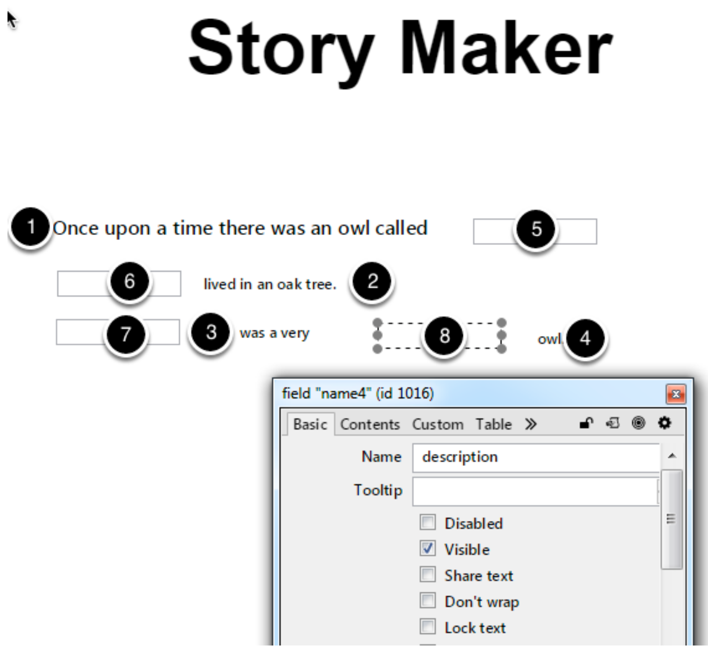
Drag a field from the Tools Palette and place it after the label field you just created (1).

Open the Property Inspector for the new field

- Set the **name** to "name1" (2)

- Set up any display properties you want for this field

Setting up the the story fields - 3

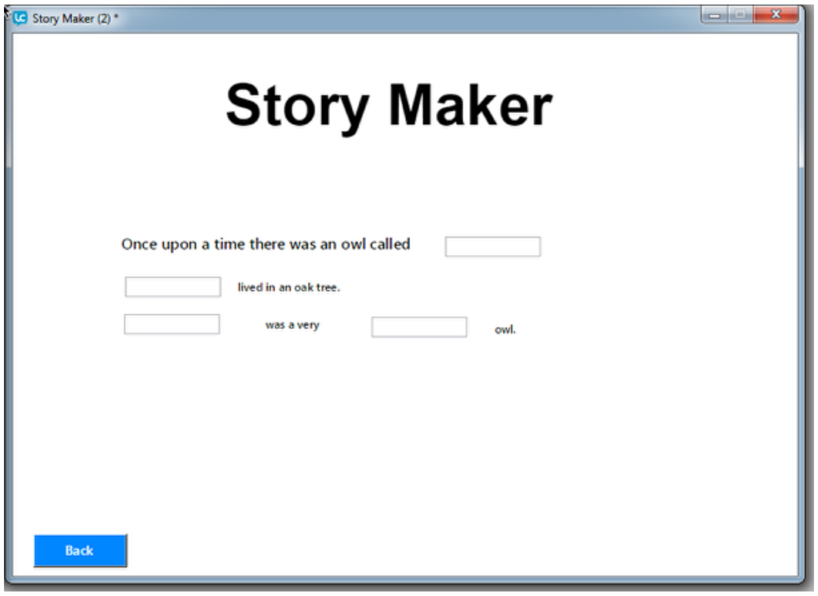


Now repeat this to build up the rest of the story. Give the fields descriptive names as we will be referring to them in code.

My story has 4 label fields, "part1" (1),"part2" (2),"part3" (3) and "part4" (4), 3 fields that will display the owl's name, "name1" (5),"name2"(6) and "name3" (7) and one field that will describe the owl, "description" (8).

You can set up your story however you like, this is just an example.

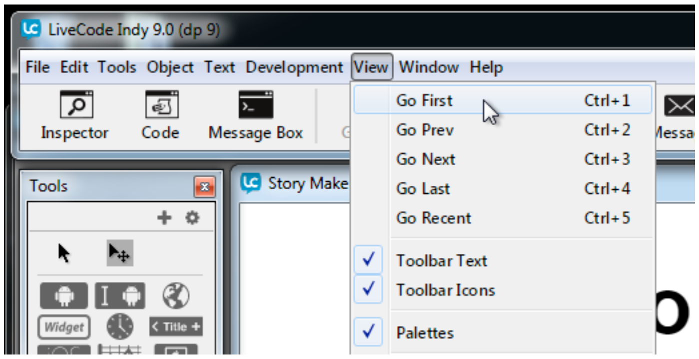
Adding a button



The last thing we need to add here is a button that will take up back to the introduction screen.

As before drag a button onto the stack, name it "Back" and set up it's display properties using the Property Inspector.

Coding the application



Now we are ready to start making the application work. Go back to the first card using the **Go First** option in the View menu.

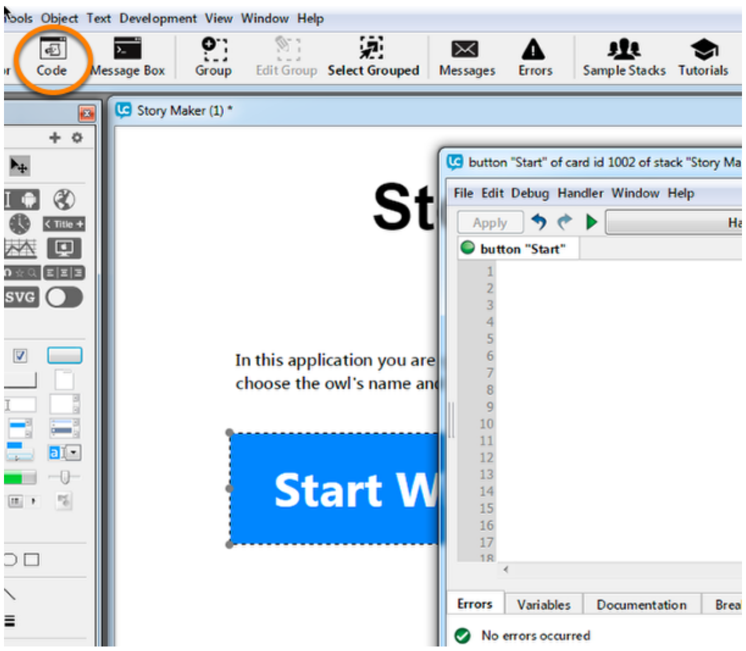
You should now see the Introduction card again.

Events

LiveCode is an event driven language, this means that it waits for a user interaction and informs us when one takes place.

We create our application by deciding which events to respond to and creating **handlers** with names that match the event. The code in the handler is executed when the matching event occurs.

Adding code to the Start button



In this case we want to respond when the user clicks the Start button. The message we are interested in is mouseUp.

Select the button and open the Code Editor by clicking the Code button in the Menubar.

The Start button code

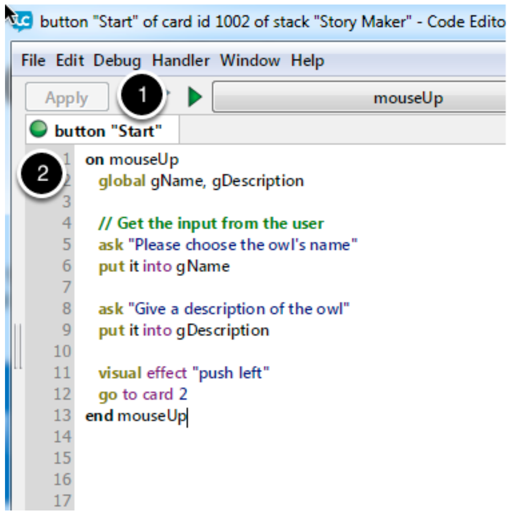
In the Start button we want to ask the user to choose the owl's name and describe it, we want to store their answers and then we want to go to the Story card. We will use global variables to store the name and description and the **ask** command to get input from the user.

Add the following code to the code editor:

```
on mouseUp
   global gName, gDescription

   // Get the input from the user
   ask "Please choose the owl's name"
   put it into gName

   ask "Give a description of the owl"
   put it into gDescription

   visual effect "push left"
   go to card 2
end mouseUp
```

The go command changes cards and the visual effect command specifies a screen transition to be used when changing cards.

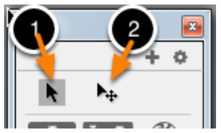
Compiling the code


button "Start" of card id 1002 of stack "Story Maker" - Code Edito
File Edit Debug Handler Window Help
Apply
1
mouseUp
button "Start"
2

```
on mouseUp
  global gName, gDescription

  // Get the input from the user
  ask "Please choose the owl's name"
  put it into gName

  ask "Give a description of the owl"
  put it into gDescription

  visual effect "push left"
  go to card 2
end mouseUp
```

Compile the Code by clicking the **Apply** button (1), the indicator should turn green to show there are no errors(2).

Testing



LiveCode allows you to switch instantly between Editing and Testing.

You do this in the Tools Palette by choosing **Run** or **Edit** mode.

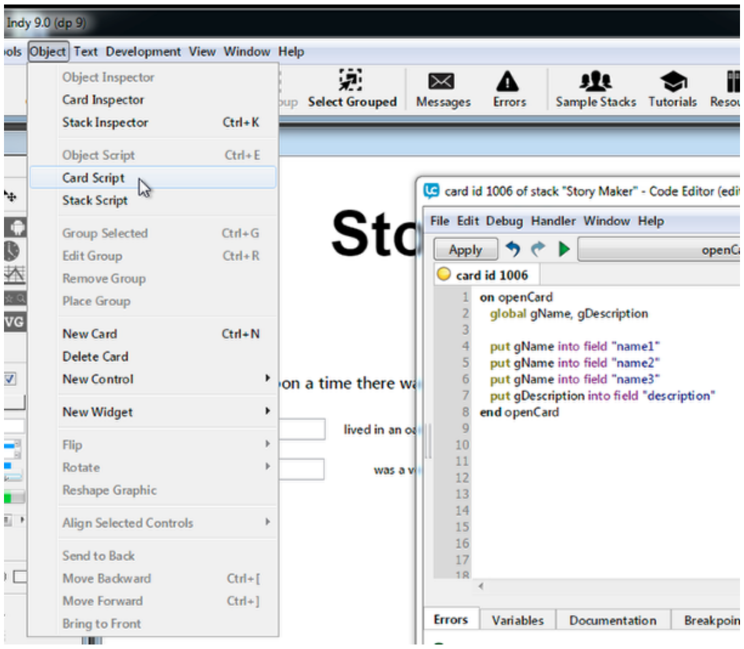
Run mode allows you to interact with the application.

Edit mode allows you to select and add objects and make changed.

Switch to **Run mode** and click the Start button.

You will be asked 2 questions and then moved to the second card.

Telling the Story



```
on openCard
  global gName, gDescription

  put gName into field "name1"
  put gName into field "name2"
  put gName into field "name3"
  put gDescription into field "description"
end openCard
```

When we move to the second card we want it to show the story, including the name and description chosen by the user.

To do this we add a handler to the card script which is called when the card is opened (arrived at, gone to). By handling this message we can update the card when it is opened. This message is **openCard.**

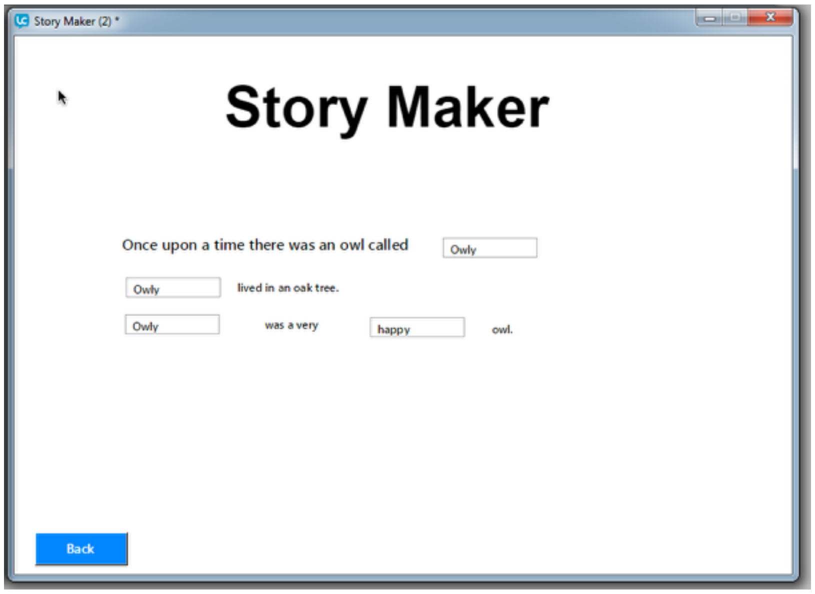
Open the **Card Script** from the Object Menu and add the following code.

```
on openCard
   global gName, gDescription

   put gName into field "name1"
   put gName into field "name2"
   put gName into field "name3"
   put gDescription into field "description"
end openCard
```

This code uses the global variables set up in the Start button and displays their values in the fields we created earlier.

Apply the code.

Adding code to the Back button



Now we want to get back to the Introduction card. To do this we add some code to the Back button.

- Select the Back button

- Open the Code Editor

- Add the following mouseUp handler

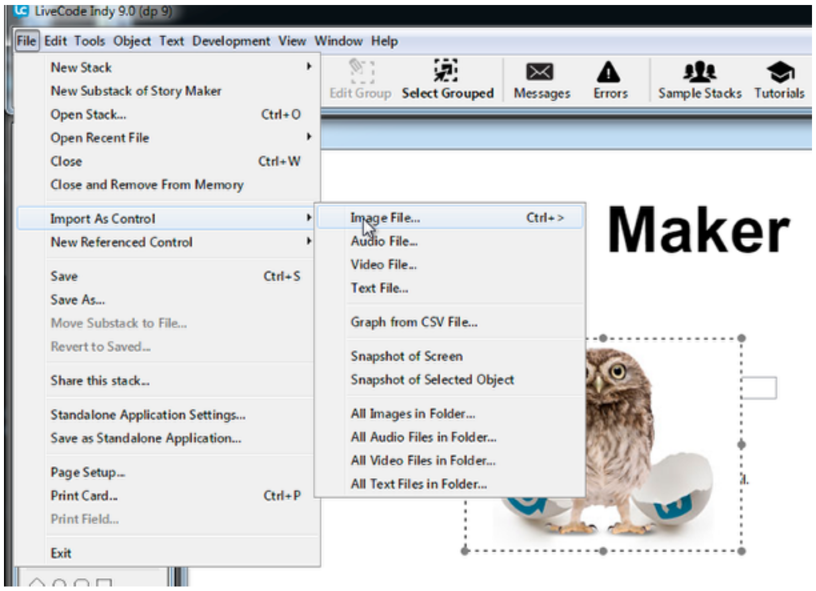
```
on mouseUp
   visual effect "push right"
   go to card 1
end mouseUp
```

- Apply the code

- Switch to **Run mode** and click the Back button. You should return to the Introduction card.

You can now click the Start button to write your story and you should see it appear.

Adding an image - 1

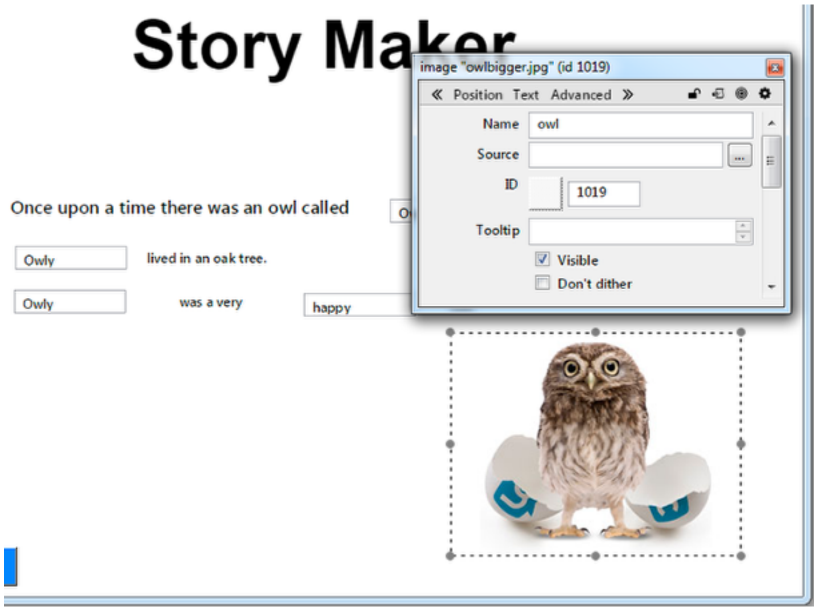


To make the application a bit more interesting we can include images.

While w are on the story card we will import an owl image, this is available in the Resources folder.

In the File menu choose Import As Control -> Image File and choose the owl image. It will appear on your stack.

Adding an image - 2



Name: owl
Source:
ID: 1019
Tooltip:
[✓] Visible
[ ] Don't dither

As with any other object type you can select the image, move it and set its properties.

- Ensure you are in Edit mode

- Select the owl image

- Move it into position

- Open the Property Inspector

- Set the **name** property to "owl"

Animating the image

To make the app more interactive we can animation the owl my moving it onscreen when the card is opened. To do this we use the **move** command which smoothly moves a control from one position to another.

Open the card script from the Object menu.

Firstly we need to move the owl image offscreen, we will do this by adding a second handler for the message **preOpenCard**, this message is sent before the user sees the card so we can do updates here. By setting the **location** property of an object we change its position, the location is the centre point of an object.

We the update the **openCard** handler to move the owl onscreen.

Add the following code to the Card script

```
on preOpenCard
   set the location of image "owl" to 1350,535
end preOpenCard
```

```
on openCard
   global gName, gDescription

   put gName into field "name1"
   put gName into field "name2"
   put gName into field "name3"
   put gDescription into field "description"
   move image "owl" to 743,535 in 2 second without waiting
end openCard
```

Apply the script.

Testing the Animation

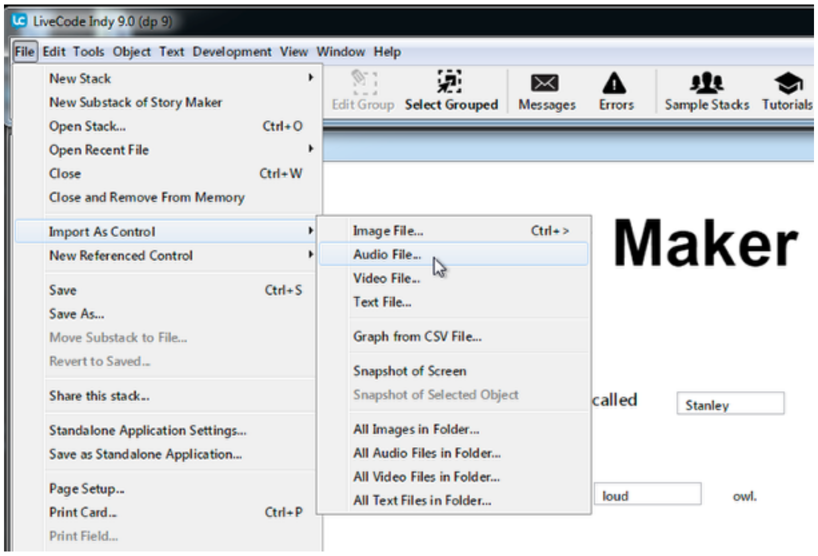
Switch to **Run mode.**

- Click the Back button to return to the Introduction card

- Create the story

- When you go to the story card the owl should move in

Adding a sound



Another nice feature of LiveCode is the ability to add sound.

Sound can be imported in the same way as images.

In the File menu choose Import As Control -> Audio File and choose the owl sound, this will import the sound and make it part of your stack.

Playing a sound

We want the owl sound to play once the image is onscreen.

To do this we handle the **moveStopped** message.

Ensure you are on the Story card, use the View menu if you need to change cards.

open the Card script from the Object menu and add the following handler, this will play the sound we just imported when the move command completes.

```
on moveStopped
   play audioclip "owl.wav"
end moveStopped
```

Apply the script, switch to **Run mode**, go back to the Introduction card and run through the story creation.

You should now be able to choose the Owl's name and description, see your story with your choices filled in, see the owl image move onscreen and hear the owl sound when the animation finishes.

Extensions

There are lots of ways you could modify this app

- Ask more questions

- Add more cards with more story

- Change the owl to a different character

- Use multiple images and multiple sounds

0 Comments