

Lab 1: Introduction to Etoys

(c) 2005, 2010 Lenny Pitt. Permission is granted to copy in whole or in part for nonprofit educational use.

Lab 1: Introduction to Etoys

We begin our journey into Squeakland's Etoys – the wonderful world of programming!

What is Etoys?

Etoys is a software environment that allows you to learn real programming. Unlike other programming languages, you do not have to type in a bunch of code, you can join together code snippets that are already written to make all sorts of interesting things happen!

Learning Etoys will teach you the basic concepts and fundamental ideas of programming through a GUI (graphical user interface) environment.

What can I use Etoys for?

You can use Etoys to make your own games, multimedia presentations, computer art, animated storybooks, computer simulations and many other things. Just wait and see!

Where can we find Etoys?

Etoys is available for free download from www.Squeakland.org.

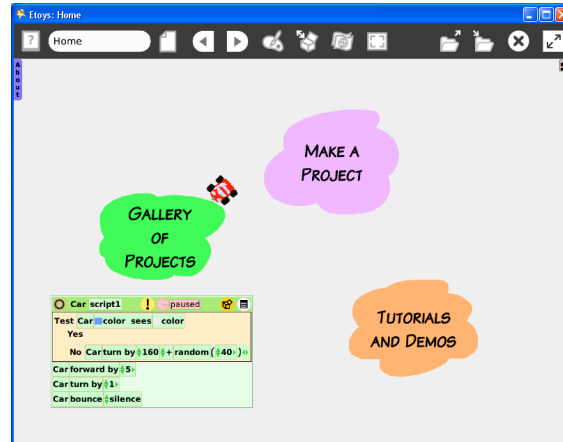
How do we use Etoys?

To start Etoys, simply double-click on the Etoys alias which is on the desktop. (It looks like a Starchild). If you do not see it there, ask for help.

University of Illinois Computer Science Department Etoys Lab Manual

Getting to know Etoys better

When you open up Etoys you see this example project with a **Navigator Bar** at the top.






This is called the **World**. Everything you do from now on will happen in this **World**! Whatever you do in Etoys is called a project. We'll refer to your work as **projects** from now on.

Navigator Bar

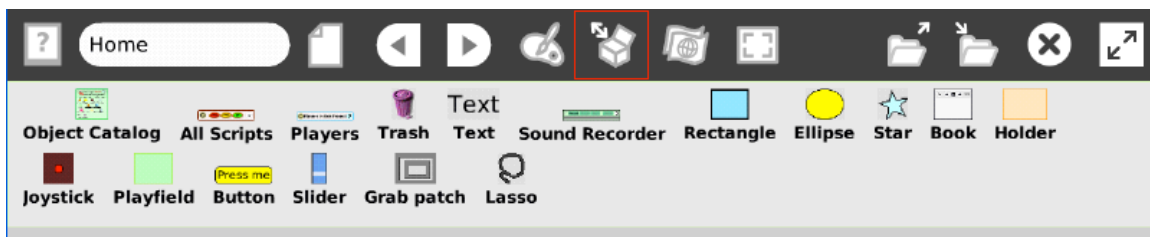
The **Navigator Bar** has many buttons:



All these buttons will allow you to navigate across different Etoys projects (done by both you and others). For now, note the **Publish It** , the **Find**  and the **Quit**  buttons. To save a project you will use the **Publish it** button. To find a project that you have saved you will use the **Find** button. To leave Etoys you will use **Quit**.

Supplies

When you click on the **Supplies** box , you get something like this:



We'll be using stuff from **Supplies** later on in our projects.

University of Illinois Computer Science Department Etoys Lab Manual

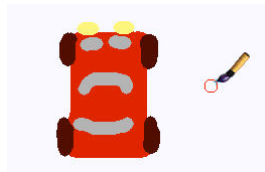
Your very first Etoys project: Operating a Car using a steering wheel.

Step 1: Paint the car

1. Click on **Make a Project**, or the page icon, to open up a new project.
2. Choose the paint brush to open up Etoys's paint palette.
3. Draw a blob – which looks like the shape of a car as seen from above.



4. If you don't like what you drew, click on **Undo** on the paint palette.
5. Choose different colors for the wheels, windows and so on.
6. When you are happy with what you have drawn, click on **keep** to save your drawing.



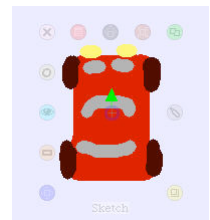
Congratulations! You have just created your first Etoys object.

Step 2: Making your car move

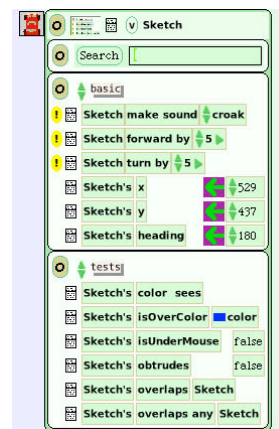
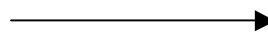
Right now your car object is just a drawing. But every object in Etoys has characteristics (properties) and behavior, just like real world things. You can manipulate the properties and behavior of the objects to make them do different things.

If you hover your mouse over the car object, you will see little ovals surrounding the car. (If you do not, then try alt-clicking (PC) or command-clicking (Mac) on the object.) These are called Handles. Through these handles, you can access the properties of your object and also modify its behavior.

For now, we will be concentrating only on the viewer handle - which is the turquoise eyeball. Click on it to reveal the viewer.



Click on the viewer handle to get the viewer



University of Illinois Computer Science Department

Etoys Lab Manual


The viewer contains property tiles that show what your car is and what it can do.

You can “collapse” the viewer against the right edge by clicking on the small tab with the picture of your car on it. To make the viewer re-appear, simply click on the tab again.

Step 3: Changing the *name* property of the car:


First, let's change the name of the object from **Sketch** to **Car**. Find the word Sketch at the top of the viewer. Click on it to highlight it and type in the word **Car**. Press **enter** to save the new name. From now on, your object will be known as **Car**.

Step 4: Getting into more action!

Look at the tiles in the **basic** category pane. You will see two types of tiles: some are preceded by a yellow exclamation point  and others are not. The tiles following an exclamation point are *action* tiles.

Clicking on an exclamation point will fire the action once. Holding down an exclamation point will ‘run’ the action repeatedly. Try driving the car around the world using the exclamation points.

Step 5: Changing the behavior of the car - manually

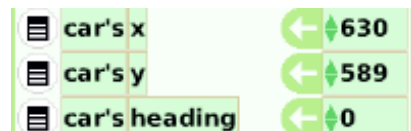
Numeric values of the tiles can be changed by either clicking on the up or down arrow on the left of the value or by selecting the value and typing in a new number and hitting **enter** once. After changing the value, click on the  again.



Try different numbers – what do you think will happen if you type in negative numbers?

Step 6: Changing the properties of the car

Tiles which are not followed by an exclamation point are *value* tiles. Each of these is followed by a green and white arrow that assigns or sets that value as the current value for a particular attribute or property of the car.



In the real world, objects have properties, such as size, color, location, direction, and so on. Every object that you draw or pull out of the supplies bin in Etoys automatically has many standard properties that tell us about the object, where it is, what it looks like, etc. Later we will see that we can create new properties for objects (for example, “car’s age”, and make the program behave differently (example – go slower) depending on the age).

University of Illinois Computer Science Department

Etoys Lab Manual

Step 7: Changing the behavior of the car - automatically

Manually pressing down on the exclamation point as you did in Step 5 to make the car move seems very inefficient and not very interesting. However, to make the car move on its own, until some condition is reached (i.e. for one minute, until the car reaches an obstacle etc) we can use *scripts*.

1. Creating a script

Scripts are created for objects by assembling tiles – in a scriptor (**script** editor). There are two ways to get a scriptor. For now, use the first option.

- (a) Drag out the tile “car forward by 5” from the viewer and drop it on the world.
A scriptor will sprout around it. You can do this with any action tile.
- (b) If you’d like to start with an empty scriptor, go to the **scripts** pane of the viewer, and drag out “car emptyscript”. (You can scroll between panes by clicking on “**basic**” and additional pane choices will appear.)




2. Naming the script

You now probably have something like this:



Click on “script1” to highlight it, and then change it by typing “drive”. When we program, we want our scripts and objects to be named things that tell us what their purpose is.

3. Running the scripts

Clicking on the exclamation mark  fires the action once – meaning the car will go forward 5 pixels . However, to make the car *keep* moving, click on the clock symbol (ticker)  on the scriptor. Notice that in the title of the script, the word  has changed to say “ticking” If you click it again, the script will stop, and it will show “paused”.

4. Experiment with different values and different scripts

Change the different values. How can you make your car go faster? Slower? Go backwards? Go around in a circle? What tiles do you need to add for these?

5. Changing the “ticking” rate.

Another way to change a script is to modify its “ticking rate”. Every script is born with the default rate of 8 ticks per second. You can speed things up, or slow them down, by holding down the mouse button over the clock icon until the following menu appears, after which you can make your selection.



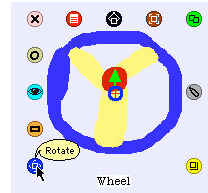
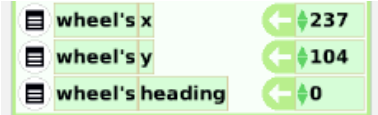
University of Illinois Computer Science Department Etoys Lab Manual

Step 8: Creating the steering wheel object

1. Paint a steering wheel.
2. Bring up the handles of the steering wheel by alt- or command- clicking on it.



3. Open the viewer for the steering wheel by clicking on the eyeball handle.
4. Change the name of the object to **Wheel**.
5. Bring up the handles of **Wheel** again. Click on the blue **Rotate** handle.

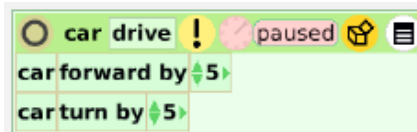


6. By clicking and dragging the blue rotation handle you can rotate the steering wheel. If you look at the wheel's **heading** in the viewer, you will see how the values change as you rotate the wheel.



Step 9: Controlling the car through the steering wheel

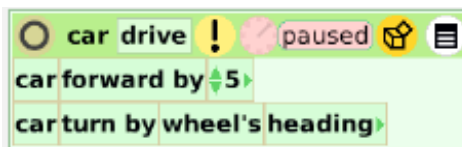
In the **car**'s scriptor, make sure you have the following tiles, as shown in the figure.



To control the car through the steering wheel, the car and the steering wheel have to be connected. Specifically, the turn of the car has to be connected to the heading of the steering wheel. Heading indicates at what angle the object is rotating. As the car moves forward, if the heading of the steering wheel is negative, it will turn left. If the heading is positive, it will turn right and if the heading is 0, then the car will move straight ahead.

University of Illinois Computer Science Department Etoys Lab Manual

To connect the heading of the steering wheel to the turn of the car,” drag the **heading** tile of the steering wheel from the viewer and position it over the “5” in the **turn by** tile of the car on its scriptor. You should end up with this:



Step 10: Drive the car using the steering wheel

You are now ready to drive the car using the steering wheel. In **wheel's** basic pane, set the value of heading to 0. Then start the car's script by clicking on the ticker in the car scriptor. Quickly bring up the handles of the wheel. Mouse down on the blue rotation angle and start driving the car!


Step 11: Using math to scale the effect of the wheel on the car

You will probably find that the car is too sensitive to the wheel. At every “tick”, the car turns by the wheel's heading. So, if the wheel is heading slightly to the right, say, 5 degrees, then, assuming 8 ticks per second, the car is spinning 40 degrees every second. Dizzying!

Click on the small green arrow just next to the word “heading”, and it will change to say “car turn by heading + 1”. You can change the “+” to a variety of other math operations, and you can change the “1” to any number. Try some of the following and determine what they do, and which effect you want in order to get the car to be less sensitive to the wheel”

“car turn by wheel's heading -5”
“car turn by wheel's heading * 5”
“car turn by wheel's heading * -5”
“car turn by wheel's heading / 5”
and so on.

Step 12: Publish the car/save the car project

We do not “save” Etoys projects, we “publish” them. In the Navigator tab, click on the “Publish” icon . A form will appear. Using the form, name the project. Follow the guidelines given by your instructor. Include descriptive information in the name. Don't use any special symbols in the name other than dashes. You need not fill out the other information; you can come back to that later.

After you click “ok” on the form, a new window will appear. Click *once* on “Etoys” (*or wherever else your instructor tells you to publish to*) which is a special folder on your computer, and then click “Save” at the bottom of the window. Wait for Etoys to finish publishing.

Congratulations! You have just completed your very first Etoys project. Now go ahead and play with the different options that we explored!!

When you are done, go back to the Navigator tab and click on “Quit”.

Lab 1 Skill Summary

Finding Etoys on the Web

Starting Etoys

Navigator Bar

Supplies Box

Alt or Command-click to get Halo

Open a viewer for an object

Name property: changing it in viewer, or on object.

Basic properties, first glimpse: x, y, heading.

Firing scripts from viewer (forward)

Changing properties in viewer (x, y, heading) by arrows or by kbd entry.

Sprout a new *scriptor* by dragging an action tile onto the world

Opening an empty scriptor by choosing emptyscript from scripts pane

Naming a script

Tickers and fire-once usage

Adding tiles to a script

Changing ticking rate

Rotate icon and heading, degrees 0 through 180 and -180 and mod 360

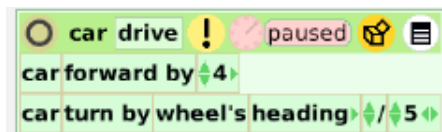
Replacing value tiles with other tiles by drag and drop (green highlight) to get objects to interact.

Use math expression to scale (divide, multiply) or change the value.
collapsing or expanding the number of terms.



Opening and closing viewers with picture tabs.

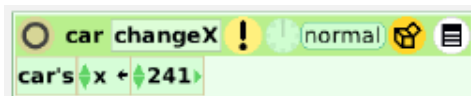
Publish and Quit

At end of this tutorial, you should have this:



Lab 1 Review Questions

1. What is a "halo" of an object?
2. How do you open a "viewer" for an object?
3. What can you view in a viewer?
4. Name five properties of any object.
5. What is a "scriptor"?
6. What is the difference between these buttons:  
7. Can you think of circumstances when you'd want to fire this script once?
How about circumstances when you'd want to start it ticking?



8. What is the "heading" of an object? What does it mean if a ticking program changes the heading so that it increases by 90? If you were riding on the object, what would you say happened?
9. How can you make an object move backwards?
10. Chelsea wants to program a "hot air balloon" to go up and down as she drags a picture of a sandbag up and down. Can you describe how she can do this in Etoys? Be real specific, because she hasn't seen Etoys before. [Challenge: how would you make the balloon go up and down much further than the sandbag?]