# PsyScript tutorial 1: Face

This tutorial covers the following subjects:

- the user interface
- image stimuli
- pausing for a fixed time
- waiting for a click
- logging results

## Prerequisites

You should not attempt this tutorial without a basic familiarity with OS X including ...

- understand what files and folders are
- understand what the Dock is
- know how to move a window around, change its shape, and close it
- know how to make a new file in TextEdit, save it somewhere, and open it again
- know what the three coloured buttons at the top left corner of a window do

The prerequisites are related to one another and a simple half-hour session should be enough to familiarise yourself with them. You may then want to take a break before proceeding with this tutorial.

## Preparation

The materials needed for this tutorial are included with the PsyScript distribution package. Before you attempt to run this tutorial make sure you have done the following:

- Make sure you have a copy of the PsyScript application on your hard disk somewhere.
- Take a copy of the folder 'tutorials/face' (included with the PsyScript distribution) and put it somewhere where you can make changes to it, probably inside your 'Documents' folder.

You're going to be starting PsyScript a number of times during this tutorial so you might want to put an icon for it in the Dock.

## Files and folders

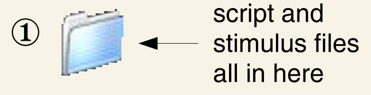
There are three types  of file associated with PsyScript:

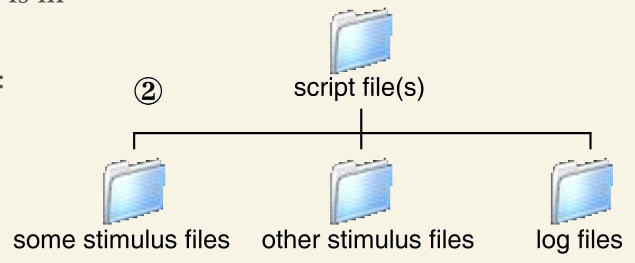
- the stimulus files – I prepared these for you: one text file and some picture files
- the script file – you'll make one of these by following this tutorial
- some log files – PsyScript creates these when you run an experiment

PsyScript works best when all its stimulus files are either

- in the same folder as the script file or
- in a folder *inside* the folder the script file is in

Here are two good arrangements for files:

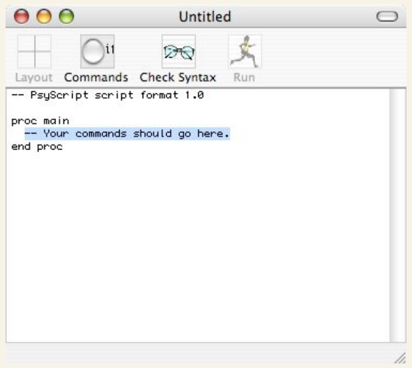




For this tutorial you'll use the first arrangement: everything in one folder.  Copy the folder called 'face tutorial' from the 'tutorials' folder that came with PsyScript into a folder you can change.

## Making a new script

Now you're ready to make a script. This is pretty easy since PsyScript does it for you. Just start PsyScript up and you should see a window showing you a short script:



```
-- PsyScript script format 1.0

proc main
  -- Your commands should go here.
end proc
```

Looking at the screen you should recognise almost everything you can see: PsyScript works just the same way as TextEdit except for the four buttons you see near the top of the window. If you know how to use TextEdit, you know how to use almost all of PsyScript. The rest of this tutorial consists of teaching you to use those four buttons.

## Saving your new script

Because your script hasn't been saved yet, two of the four buttons you see above the it can't be used. So before you start experimenting save your script. Any way of saving files in TextEdit works in PsyScript. Use one of them now to save the script in the folder you made by copying the one with the stimulus files. Call it 'tutorial 1' or something. PsyScript will automatically add an extension onto the end of the filename. Once you've saved it, all four icons will be enabled.

From now on, you're going to be editing and running your script. Your five line script which does nothing, will get longer and do something.

## Adding some commands

The first commands you're going to put into the script are commands for putting some images on the screen. Normally you'd do some typing, replacing the

```
  -- Your commands should go here.
```

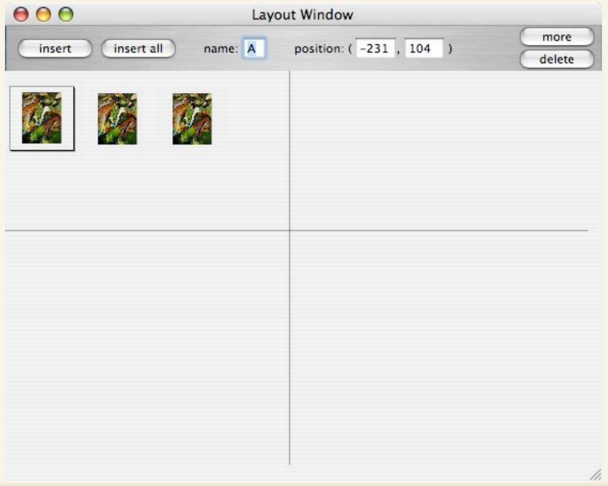
line with commands which display images. To do that you'd have to learn the syntax for the image commands but the first of the four buttons will save you the trouble by writing the commands for you. If you rest the mouse cursor over the first button (without clicking it) you should see a little yellow tag appear which says 'Layout Helper'.

## Setting up a layout

Just before you click the 'layout' button, we want to make sure the commands appear in the right place. Make sure the line which reads

```
-- Your commands should go here.
```

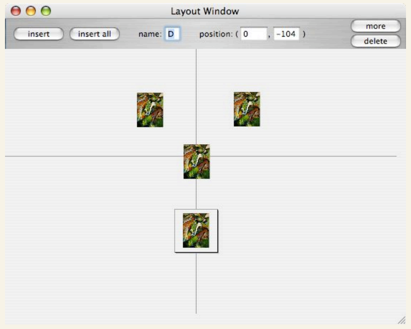
is selected. Your window should look like the one shown a couple of pages ago (except that your selection highlighting colour may not be blue). Now click the 'layout' button and you'll see this:



You're going to arrange things on this window and then PsyScript is going to write the equivalent commands into your script for you.

This window is currently showing three empty 'cells'.  A cell is something you can show an image in.  You can have up to 26 cells, named 'A' to 'Z', on your display at one time.  You can load pictures into a cell, make a cell visible and hide it, and move a cell around the display.   For now we're going to keep it simple: use four cells to show two eyes, a nose and a mouth.

At the moment you can only see three cells and you want four.  Click the 'more' button to give you one more cell.  You should be able to click on each cell and, looking at the top line, see its name and position.  Positions are relative to the middle of the screen: (0,0) is right in the very centre of the display.  Drag the cells around and watch the numbers change, or type different numbers into the boxes and watch the cells move.  End up with the four cells in roughly the right positions for A, B, C and D to be left eye, right eye, nose and mouth respectively:



name: D  position: ( 0 , -104 )

Don't worry about positioning the cells precisely, just get something that looks vaguely like a face. Once you've done that, hit the 'insert all' button and you should see some lines written into your script. If you can't see that bit of your script window, move the windows around so that you can but don't close the layout window. My lines were:

```
move cell A to (-84,94)
move cell B to (82,92)
move cell C to (-2,0)
move cell D to (-2,-82)

show cell A
show cell B
show cell C
show cell D
```

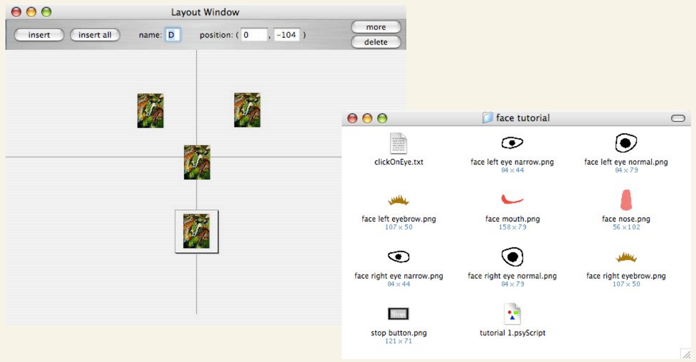
The numbers in brackets are the x and y co-ordinates of the middle of the cell, relative to the middle of the display. (-2,0), for example, indicates a place just to the left of the middle of the display. You can load whatever sized image you like into each cell but the middle of the cell will stay in the same place unless you use the `move cell` command.

Those are the commands that you'd use to move four cells to the positions you had them in the layout window. Your numbers may be different, or the lines may appear in a different order. Neither matter. In fact, we're going to delete that text anyway since we can get the layout window to do even more for us. So highlight those nine lines and delete them (we're going to replace them) and click back on your layout window again.

## Showing images in cells

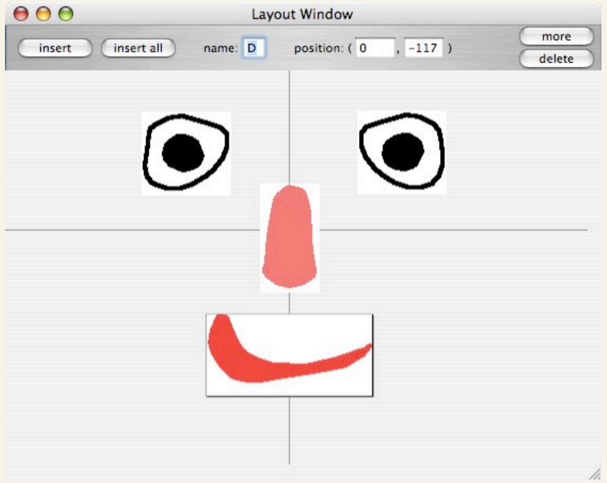
So far you've moved some cells about so they're in the right position, but you haven't said which pictures you want shown in them. You can make the layout window do this for you too but you'll need to be able to see both the layout window and the Finder window showing your stimulus files.

You may have to move your windows about a bit to make this easy. You need to see both the layout window and the Finder window showing your stimulus files. The script window doesn't matter for the moment. Depending on how big your monitor is, you might end up with something like this:


Layout Window
insert
insert all
name: D
position: ( 0 , -104 )
more
delete
face tutorial
clickOnEye.txt
face left eye narrow.png 84 x 44
face left eye normal.png 84 x 79
face left eyebrow.png 107 x 50
face mouth.png 158 x 79
face nose.png 56 x 102
face right eye narrow.png 84 x 44
face right eye normal.png 84 x 79
face right eyebrow.png 107 x 50
stop button.png 121 x 71
tutorial 1.psyScript

Now you can see everything you need for the next step. It's easy to put images in the cells you've laid out: just drag the image file onto the cell. You should be able to drag four files:

```
face left eye normal.png
face mouth.png
face right eye normal.png
face nose.png
```



I've supplied stimuli images in PNG format but many different formats work: PICT files, JPEGs, GIFs, PNG, TIFF, even PDF files should all work. If you have any difficulty, make sure that the filename extension (the bit after the '.' in the filename) is appropriate for the format the image is in.

If you drag the wrong one, drag another one over on top of it. Even after you've put images in the cells you can move the cells around, add more cells or delete some which already exist. When you're finished you should have something similar to the picture shown above.

Once you're happy with your layout, hit the 'insert all' button again.  This time I got

```
move cell A to (-84,94)
load image face left eye normal.png into cell A
move cell B to (82,92)
load image face right eye normal.png into cell B
move cell C to (-2,0)
load image face nose.png into cell C
move cell D to (-2,-82)
load image face mouth.png into cell D

show cell A
show cell B
show cell C
show cell D
```

These 12 lines should have replaced the earlier four lines.  As you can see, the button generated three commands for each cell: first position each cell in the right place and load the right image into it.  Then, once everything is set up correctly, quickly make each cell visible.

You need all three types of command to make even one image visible: you must say where it should appear, you must say what file the image is stored in, and you must say that the cell should be visible.  You can do those things in any order but you must do them all.

If you're happy with the code produced, you don't need the layout window any more so you can close it.  We're finished with that window for this tutorial.  And now's a good time to save the changes to your script, so do that.

## Finishing your first script

You're nearly ready to run this first script but you won't be able to see much without one extra command. The script will draw a face but then it'll immediately end: you won't have any time to see the face. So we're going to add one extra command just before the end:

```
wait for 5 seconds
```

it should be fairly obvious what this does. Put it after a blank line to separate it from the lines which set up the face and your script should end like this:

```
  show cell D

wait for 5 seconds
end proc
```

actually, your 'wait' command should be indented a little to show it's part of the main procedure (started by the line 'proc main' and ended with the 'end proc' line). You can make PsyScript do this for you by hitting the third button, the one with glasses on. This is the 'check syntax' button. It looks through your script to see if you've made any obvious errors, then it makes sure it's nicely laid out. If you hit that button now you should see your line move in a couple of spaces.

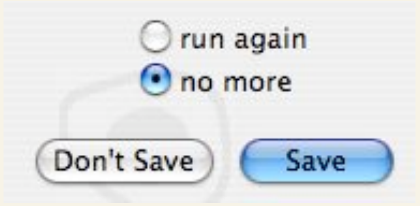
## Running the script

Your script should now end

```
  show cell D

  wait for 5 seconds
end proc
```

and you can now run it (or, to be precise, you can run the 'main' procedure) by hitting the fourth button, the one with a picture of a runner. You should see a picture of a face build up and display for five seconds, then you should see a panel with lots of fields and buttons. You can ignore almost all of it at the moment, you're only interested in the lower right corner:



Choose whether you want to run the script again or not, then hit the 'Don't Save' button.

Congratulations. You've learned how to use three of the four buttons and you've run your first script. From here on, it's all minor details. Now's a good time to take a break.

## Waiting for a response

So far we've just shown one type of stimulus to the subject. That's not a very useful type of experiment to do: the subject didn't get a chance to respond to the stimulus. That's what we'll fix now. The two simplest types of response PsyScript can log are clicking an image on the screen and pressing a key. In this tutorial we're only going to look at using the mouse to click on an image but the commands for handling keys on the keyboard are similar to the ones we'll use.

First we'll put the response command into the script. We'll worry about logging what was done later. For now, just find the line in your script that ways

```
wait for 5 seconds
```

and change it to

```
wait for a click
```

You can perform a syntax check on that if you like (if you don't, the 'run' button will do it for you before it runs the script) then run it. Previously, the script showed the face then paused for five seconds. Now it will pause until you click on one of the images, then continue. Simple enough ?

So the form

```
wait for a click
```

waits for a click on any of the images showing.  It ignores clicks on the background (the area around the images).  There's another form of this command: change the one you have in the script now to

```
wait for a click in AB
```

The difference here is that this one pays attention only to a click in cell `A` (left eye) or cell `B` (right eye).  It will ignore clicks on the nose and mouth, as well as ignoring clicks on the background.

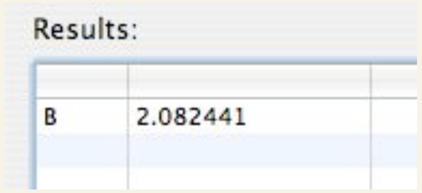
Now we've established that the `wait for a click` command waits for a click, we're going to start logging the results.  Believe it or not, you do this with the 'log' command.

## Logging the response

Underneath the 'wait' command add the three following lines:

```
log $lastClick
log $lastClickTime
log $return
```

(those symbols are dollar signs: shift-4 on most keyboards). Click the 'Check syntax' button and they should indent correctly below the 'wait' line. Then run the script again and click on one of the images. This time, for the first time, the log grid (the thing at the top of the results panel) won't be blank. Mine showed this:



```
B       2.082441
```

Looking back at the two commands you inserted you can tell that the first thing, the '`B`', is the `$lastClick` thing: the last cell that was clicked. The second thing, '`2.082441`', is `$lastClickTime`: the number of seconds PsyScript had to wait before I clicked on a cell.

The table those things are shown in reacts like a normal table under OS X: you can make the columns wider or narrower and change the order by dragging the column header left or right.

If you make the second column wider by dragging the right side of the header above it to the right you'll find out that the timing measurement is actually far more precise than three decimal places. However, there's no way for me to tell how accurate your computer is at measuring time. It might be accurate to thousandths of a second or to millionths of a second or perhaps only to tenths of a second. For the purpose of analysing your results, you should be rounding or truncating your timing figures and possibly quoting confidence intervals, whatever fits your experiment and methodology.

Congratulations again: you have now shown stimuli and logged the results. That's all you need for psychology experiments. From here on, it's all minor details.

When you're ready, proceed with part II of the Face tutorial.