


1 Review and preview

- (1) Previous handouts:
<http://www.linguistics.ucla.edu/people/grads/aalbrigh/psyscope>
- (2) Review of topics covered last time
 - creating a text event
 - using stimulus attributes to change the font, size, etc. of the text
 - using event attributes to change the duration of the event to wait for a key press
 - using lists of stimuli
- (3) Topic for today: a ratings task with sound and text
 - preparing sound files for use in psyscope
 - including sounds in a trial
 - recording ratings entered with the keyboard
 - recording reaction times

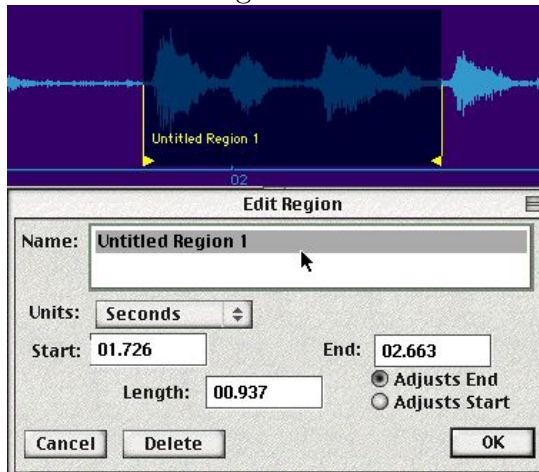
2 Preparing sounds for PsyScope

- (4) The short answer: SoundEdit 16 files
- (5) The longer (=Adam's idiosyncratic) answer
 - digitize long chunk using SoundRecorder
 - divide into individual files using Peak
 - convert to SoundEdit using the SoundEdit automator
- (6) Digitizing using SoundRecorder (installed on Isis)



- Controls
 - Source settings: *Built-in, Sound In*
 - Format settings: *44100 Hz, 16bit, mono*
 - Save as: “Export Sound to AIFF”
- (7) Breaking up the file with Peak
 - open AIFF file in Peak
 - select a piece (use space to play current selection)
 - create a new region with command-shift-R, or the region icon: 

- name the new region and click OK:

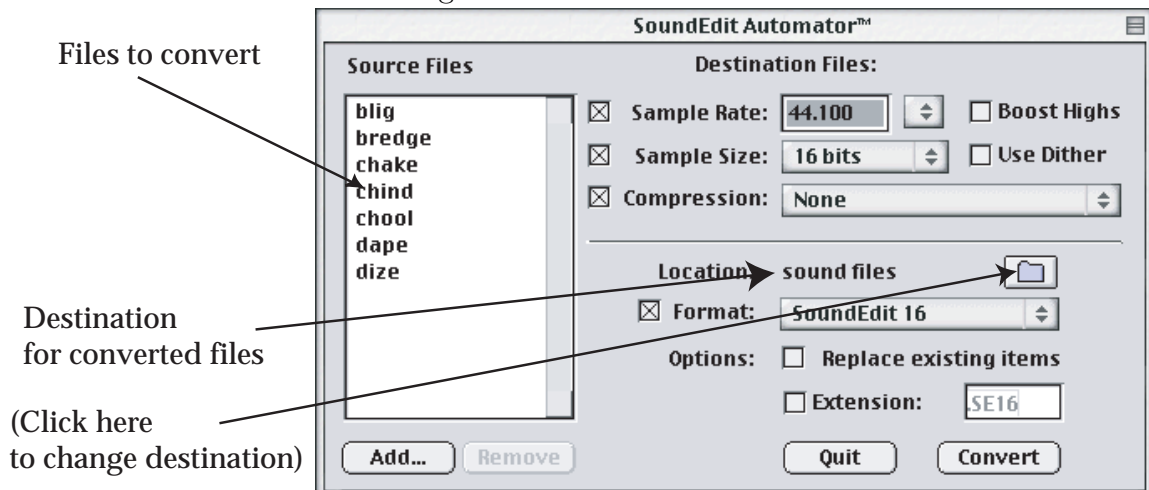


- export regions to separate files by selecting the entire file, and choosing “Export regions” (File menu)

(8) Three warnings about Peak:

- *always work from the hard drive, never from a ZIP disk. Particularly never from a PC-formatted ZIP disk.* (Hundreds of Amanda’s files died to bring you this message)
- *when you launch Peak and it says it “could not be found,” this is its cryptic way of telling you it needs more memory.* Quit all other applications, or restart if necessary.
- *if you change your mind while creating a region, use DELETE, not CANCEL* (this is a UI glitch in Peak)

(9) Convert to SoundEdit files using the SoundEdit Automator:



(10) A seemingly simpler, but ultimately more time-consuming method

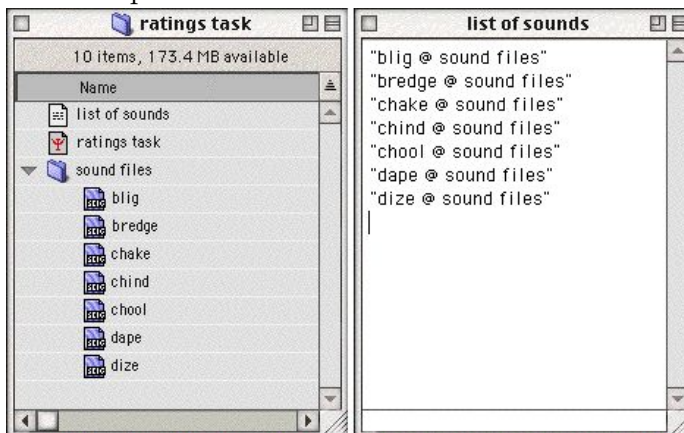
- open up the large file in SoundEdit to begin with, copy little chunks and paste into new documents, save and name them

- (11) Even slower, but possible:
- digitize in tiny chunks using your favorite PC-based software, save as .WAV files, and convert with the SoundEdit Automator

3 Playing your sounds in PsyScope

- (12) Remarkably easy: create a sound event (🔊)
- (13) Playing a single sound file: edit the stimulus attributes for the sound event, setting the “File:” attribute to the file you want to play
- (14) Notice that under the event attributes for the sound event, the duration is automatically set to “*sound[]*,” meaning the event lasts until the end out of the sound file
- (15) Playing sounds from a list: just make a text file with all of the names of the files
- you can keep all your sound files in a subdirectory called something like “*sound files*.” In this case, you must include both the path and the filename in your list. The format is: “*filename @ :directory*.”
 - note that “*filename @ :directory*.” contains spaces, so you must enclose it in double-quotes in your list file!

- (16) An example:










4 Putting it together: a simple ratings task

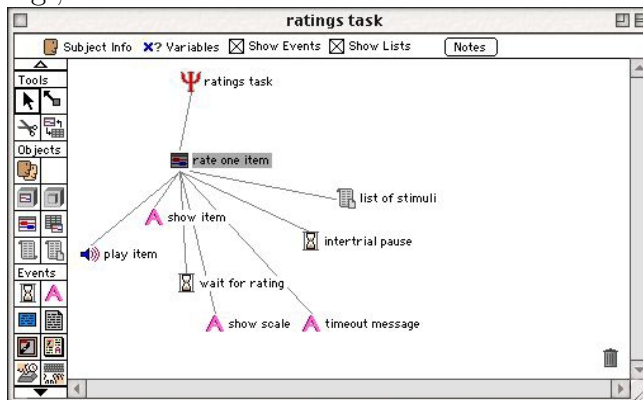
4.1 Format of the task

- (17) Format::
- play a sound file, and display some text simultaneously
 - after sound is over, display a ratings scale
 - trial ends when subject presses a number 1-7


- trial times out after 10 seconds, displaying a brief message saying so
- brief pause (2 secs) between each item

(18) Some events we will need:

-  a template for the trial
-  a sound event (play the stimulus)
-  a text event (simultaneously show the stimulus orthographically)
-  a time event (wait for subject to enter rating)
-  a text event (ratings scale, to remind subject)
-  a text event (informing subject that they didn't respond in time)
-  a time event (intertrial interval)
- e.g.,



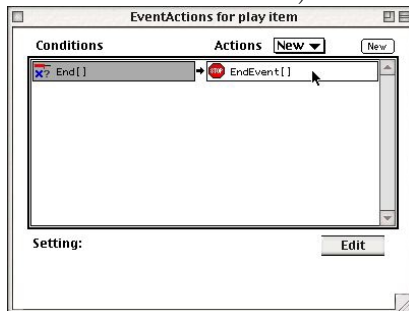
4.2 Sequence of events in the trial

- (19) stimulus text will begin at same time as sound event (0 msec after start of the sound event)
- (20) stimulus text will disappear when the sound event ends
- set duration of text event to “self-terminate” ()
 - now add an “action” to the sound event, to kill the text event when the sound is done playing
 - edit event attributes of the sound event
 - under “Actions” choose “Set to:”

- click “*New*” to create a new EventAction:



- double-click “*Never*” to change the conditions; in this case, we want to do something when the sound ends, so select “*End[]*”
- now under the “*Actions*” pull-down menu, choose “*EndEvent[]*” (we want to end the text event)



- finally, double-click the “*EndEvent[]*” action and set its target to your text event

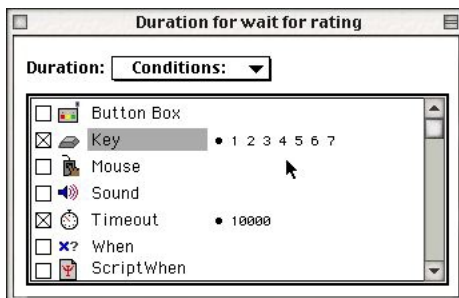
- now when the sound event ends, it will tell the text event to kill itself too

(21) the time event to wait for a response will begin as soon as the sound event finishes (i.e., “*Start 0 msec after end of event: play item*”)

(22) the time event will be ended in one of two ways:

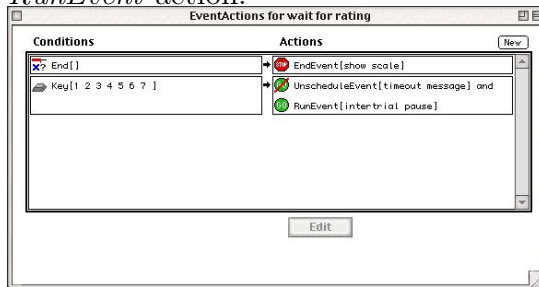
- a keypress with a rating (1-7)
- a timeout after 10sec

(23) edit the duration attribute of the timer event, to end under these two conditions:



(24) Recall we also wanted a text event, showing the scale while the subject is deciding on a rating. Give your scale text event some text, and make it begin and end at the same time as the timer event:

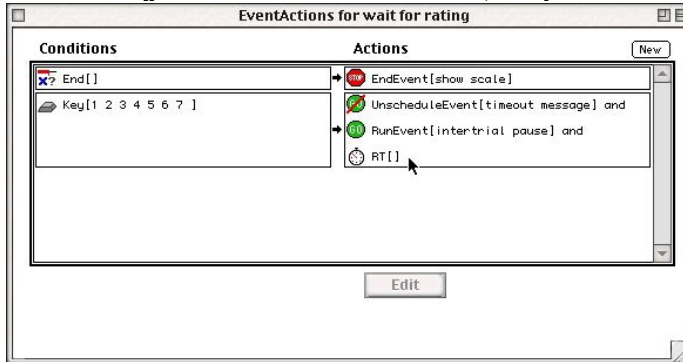
- this text event will begin 0 msec after the beginning of the timer event
 - make it self-terminating (it does not determine its own duration)
 - add an action to the timer event: under the condition of “*End[]*”, it will do an *EndEvent* to end the scale text event. (see demo)
- (25) How can we handle the timeout for rating items?
- if the subject does nothing, we want the timeout message to appear, and then to go on to the next trial
 - if the subject enters a rating, we don’t want the message to appear
 - thus the *default* will be to have the message appear – it’s easier to cancel the message when something happens (apply like fire) than it is to call the message when nothing happens (apply like water)
- (26) So: give the timeout message some text, a style, and a duration, and schedule it to occur immediately after the “wait for a response” timer event
- (27) The intertrial pause (another timer event) will come immediately after the timeout event
- (28) Now: cancelling the timeout message when a rating is entered
- this is another action for the “wait for rating” timer event to perform
 - in this case, the action occurs only under particular conditions: a keypress from 1 to 7. click “New” on the far right to create a new set of conditions, and set it to a keypress of 1 through 7
 - now specify the actions to be triggered by the keypress. The timeout message hasn’t occurred yet, so we can’t kill it by just ending it. We have to *unschedule* it.
 - however, we do still want the intertrial pause to take place – it was scheduled to occur after the timeout message, so now we must call it manually with a *RunEvent* action:



4.3 Collecting the responses

- (29) Running experiments is useless if you don’t collect the data! Fortunately, this is very easy in Psyscope.
- (30) In this case, we want to know which key was pressed, and how quickly

- the keypress comes during the “wait for keypress” event
 - it already triggers some events (de-schedules the timeout message, runs the intertrial pause, ends the event)
 - we just need to add one more action: a $RT[]$ (reaction time) action
- (31) Edit the event attributes for the “wait for rating” timer event. Under “actions”, add a new $RT[]$ action to the keypress ($Key[1\ 2\ 3\ 4\ 5\ 6\ 7]$) condition:



- (32) That’s all! now PsychoPy will record the reaction time and the keypress to the data file. (more on those another time)

5 Exercises

- (33) [0] idiot-proof your ratings task to survive the caps-lock key accidentally being pressed
- (34) [5] add a instructions screen at the beginning and a thanks screen at the end
- (35) [5] change the trial template so the stimulus text remains on the screen until a rating is entered (or the trial times out). (hint: you won’t need the timer event for waiting for a keypress any more)
- (36) [10-15] add a training phase, in which the task is explained and the subject practices entering ratings. the items in the training phase should not be in random order, but the items in the real task should be randomized.
- (37) [30+] add a feature to allow the experimenter to pause the experiment mid-way through, using some secret key/button not likely to be activated accidentally by the subject. when the key is pressed, a message should appear saying something like “click to continue”, or perhaps giving the user a choice about what to do (1 to quit, 7 to continue). when the user continues (unpauses), the current trial should be repeated.