PsyMSc4 Kog - Praxismodul

Python for Psychologists

Dominik Kraft
WS 20/21

Create dialogue boxes

What?

Creates a little dialogue box that asks for certain inputs from the experimenter.

Why?

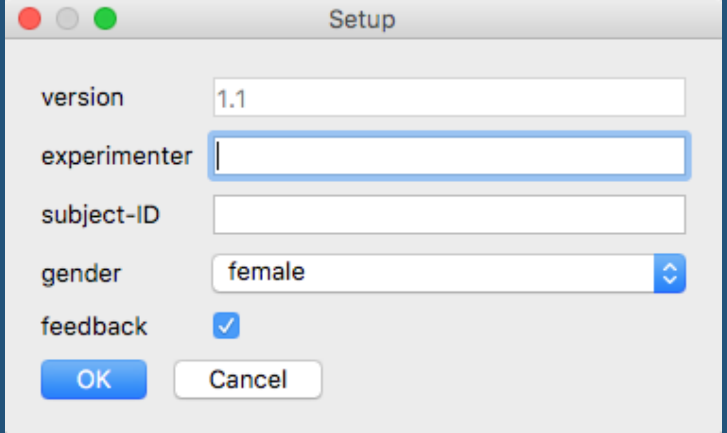
Experiments usually need input from the experimenter:

- subject-ID
- condition

…

psychopy.**gui.DlgFromDict(**dictionary, title="title_of_the_window", order=[list_of_fieldnames_in_order], fixed=[list_of_unchangeable_fields]…**)**

- creates a dialogue box
- pass a dictionary with the following forms "allowed":
  {"field1_name":["drop", "down", "menu"]}
  {"field2_name": ""}
  {"field3_name": "pre-set value"}
  {"field4_name": True}
- important attributes:
  .dictionary --> contains the filled-in dictionary
  .OK --> returns True or False, depending on whether dialogue box was ended via "ok" button press or not



Setup

version 1.1

experimenter

subject-ID

gender female

feedback ☑

Task 1: Try to rebuild this dialogue box! Add it to the experiment script from last session

Task 2: If the cancel button has been pressed, have PsychoPy stop the experiment (to stop: core.quit())!

Set global event keys

What?

Global keys are keys that are always automatically checked for at any time in the experiment. Once, the key is pressed, ANY function you want will be executed immediately.

Why?

For example, because we need to be able to exit the experiment at any point in time.

psychopy.**event.globalKeys.add(**key, func, name**)**

- Defines new global keys
- Needs the key (as a string) and the function to run as inputs

psychopy.**event.globalKeys.clear()**

- Clears all previously set global keys

psychopy.**event.globalKeys.remove(key)**

- Remove single key settings

psychopy.**event.globalKeys**

- Returns the global keys currently set

Task 3: Define such a global key named "shutdown", that simply quits when it is hit. Clear all existing global keys before.

Logging data: Python “standard”

What?

Create a logfile to store all infos like RTs and key presses.

Why?

For analysis!

We can simply use python’s built-in commands to write data to a file. That is, the way of logging data shown on this slide is not at all specific to PsychoPy.

```
with open(logfile_name, mode=‘a’/’w’/’r’) as some_var:
        print(“some string!”, file=some_var)
```

Task 4: Create a logfile and first write the subject-ID down. After each trial, write the key press of the respective trial into the logfile (also misses as “miss”). Also, make the subject-ID part of the file name.

Logging data: the PsychoPy way

What?

Creates a logfile that (potentially) logs all relevant events.

Why?

For analysis!

Logging levels:

- ERROR
- WARNING
- DATA
- EXP
- INFO
- DEBUG

psychopy.**logging.LogFile(**f="filename", level=logging.SOME_LEVEL, filemode=["a", "w"]...**)**

- creates a new logfile that saves events
- relevant method:
  - .write("Some text message")

NOTE:

Many objects (like windows or stimulus objects) have an autoLog parameter which, when set to True, makes the logfile record all events happening with respect to this object. Response times can later be calculating by subtracting stimulus onset times and the DATA input (i.e., key press, etc.) times.

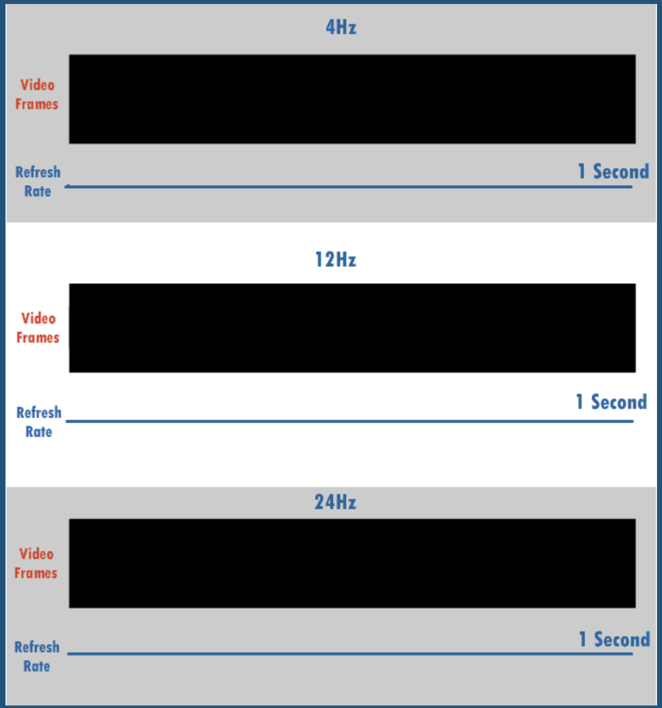
(Task 5) Create a new logfile with level logging.INFO. Also, set the autoLog parameter to True for both your window and your TextStim object. Then, run the experiment and take a look at the logfile.

Refresh rate

The refresh rate refers to the number of times per second that a display hardware device updates its buffer

Most monitors you will be working with will have a refresh rate of 60 Hz which corresponds to a screen update every 16.67 ms .

The refresh rate limits the points in time at which you can present a stimulus. For example, it will be impossible to display a stimulus for exactly 225 ms, as 225 is not a multiple of 16.67; only 216.71 or 233.38 ms are possible.



https://en.wikipedia.org/wiki/Refresh_rate#/media/File:Waveform_video_comparison.gif