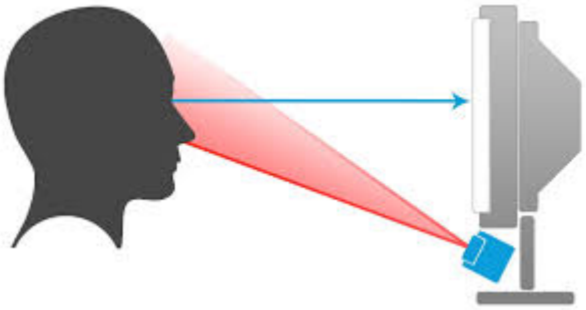
Eye tracking and Open Sesame: practical

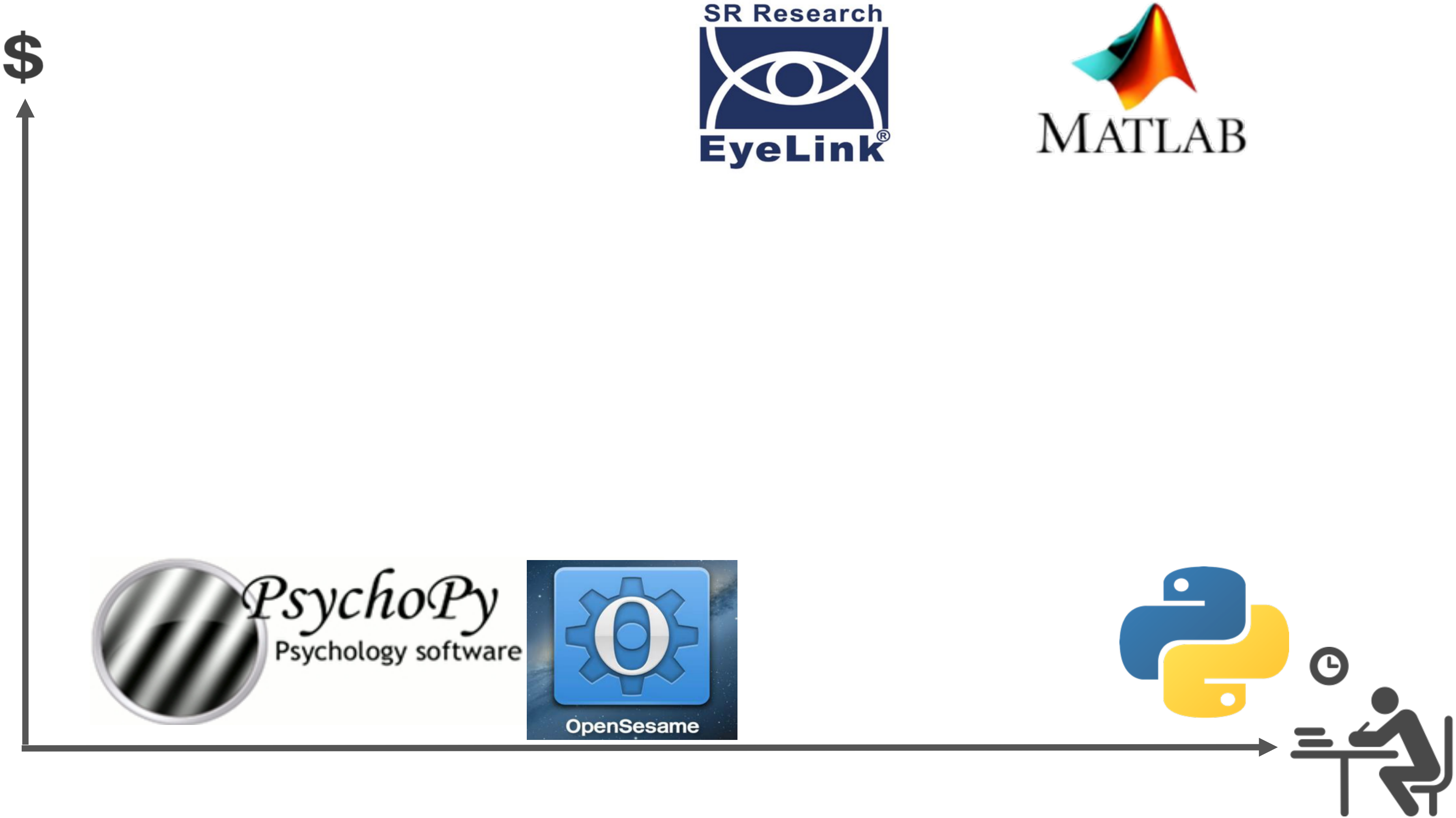
Liya Merzon

You decided to do an eye tracking experiment

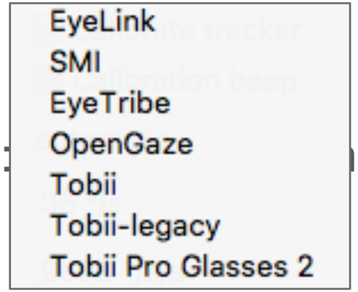


Is OpenSesame what you need?

Let’s look at available options


$
SR Research
EyeLink®
MATLAB
PsychoPy
Psychology software
OpenSesame

+

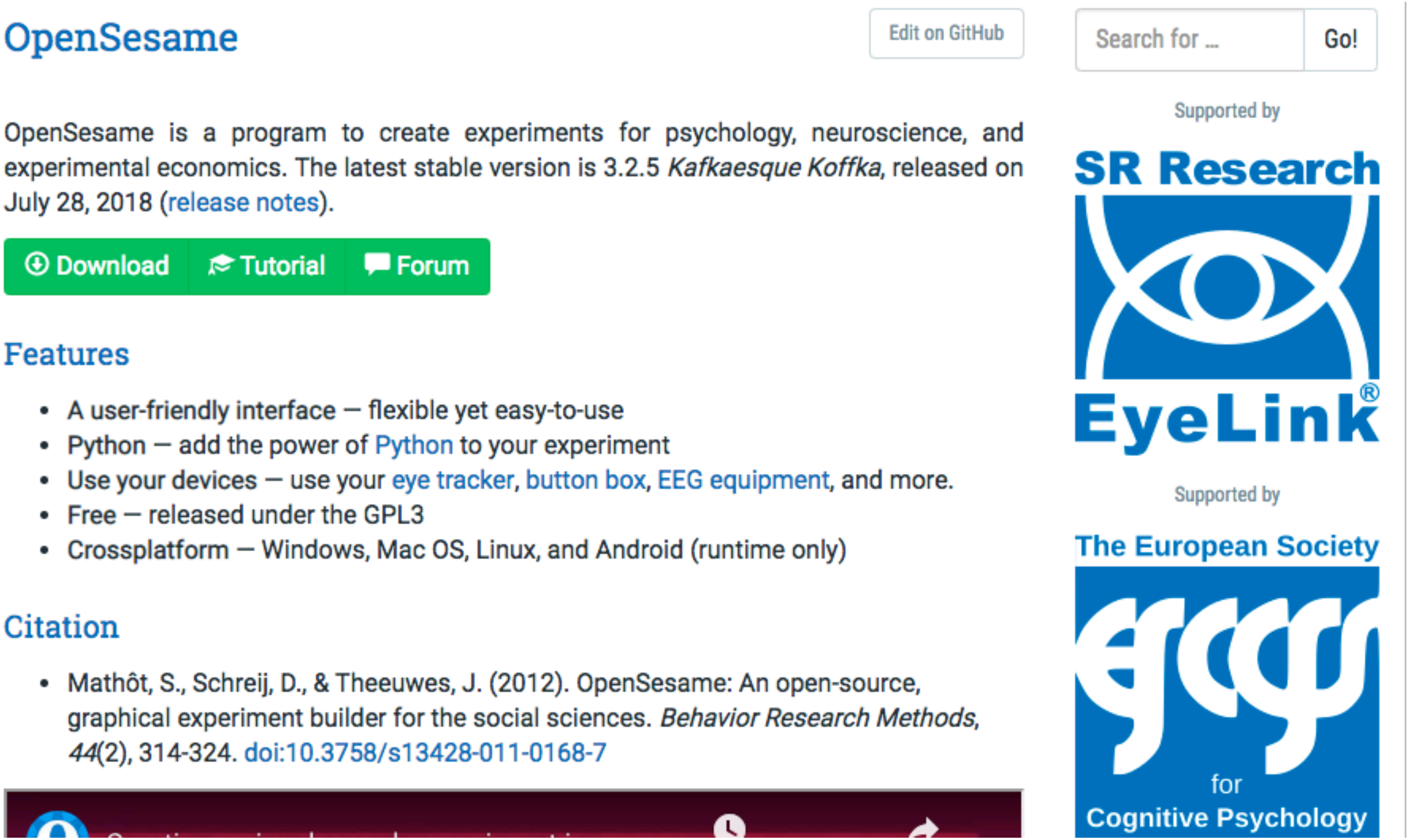


1. “OpenSesame is, and will always be, free software” © https: l/ 3.2/download/
2. Windows, Mac OS, Linux
3. Supported by SR-research (will be work with Eyelink well:))
4. Easy: has graphical interface

   ! Including graphical wrapper for eye tracking set-up

1. More complex things could be done by adding several lines of Python code
2. Well documented, a lot of tutorials, templates and examples

https://osdoc.cogsci.nl/3.2/



OpenSesame

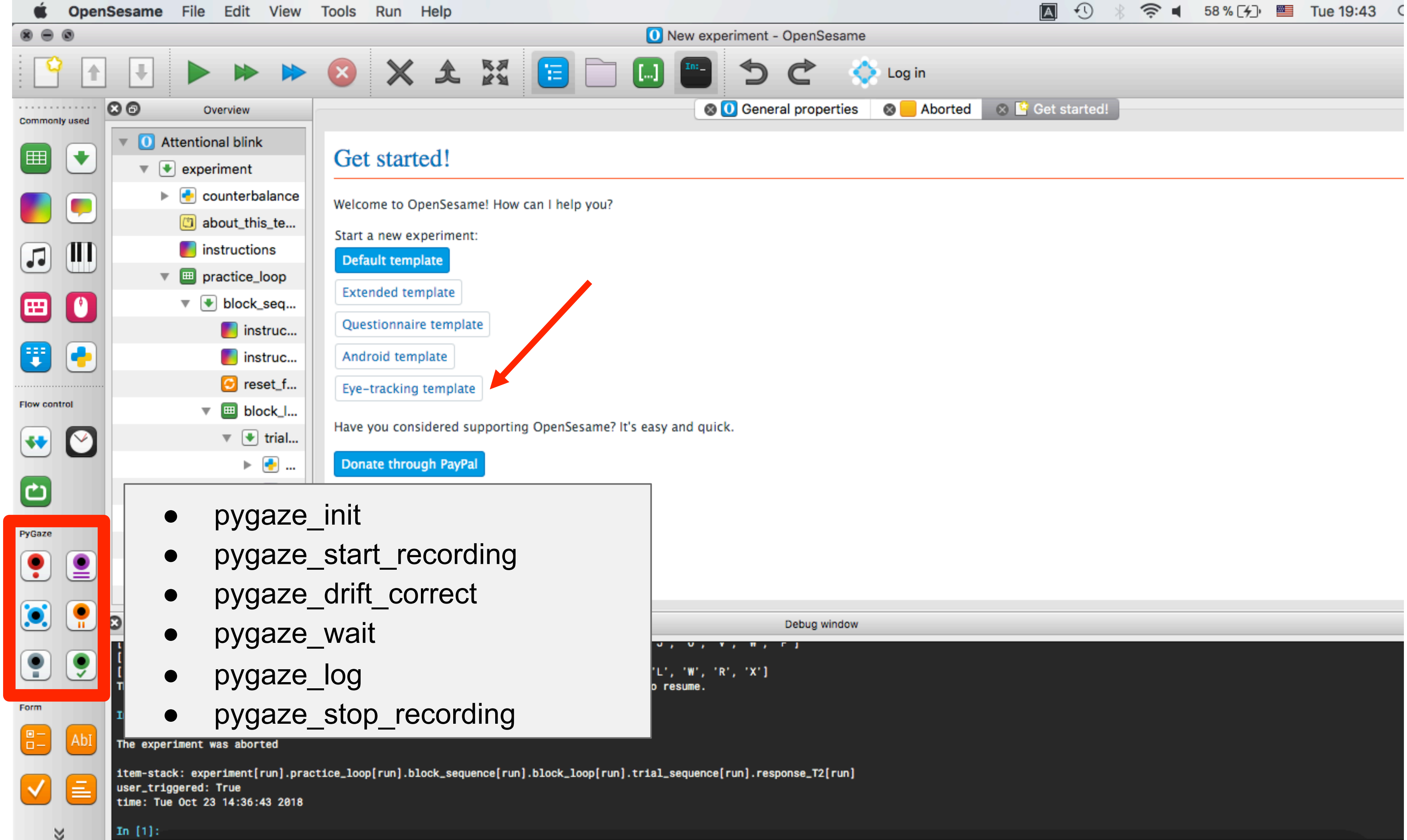
OpenSesame is a program to create experiments for psychology, neuroscience, and experimental economics. The latest stable version is 3.2.5 *Kafkaesque Koffka*, released on July 28, 2018 (release notes).

Features

- A user-friendly interface — flexible yet easy-to-use
- Python — add the power of Python to your experiment
- Use your devices — use your eye tracker, button box, EEG equipment, and more.
- Free — released under the GPL3
- Crossplatform — Windows, Mac OS, Linux, and Android (runtime only)

Citation

- Mathôt, S., Schreij, D., & Theeuwes, J. (2012). OpenSesame: An open-source, graphical experiment builder for the social sciences. *Behavior Research Methods*, *44*(2), 314-324. doi:10.3758/s13428-011-0168-7



Get started!

Welcome to OpenSesame! How can I help you?

Start a new experiment:

Default template

Extended template

Questionnaire template

Android template

Eye-tracking template

Have you considered supporting OpenSesame? It's easy and quick.

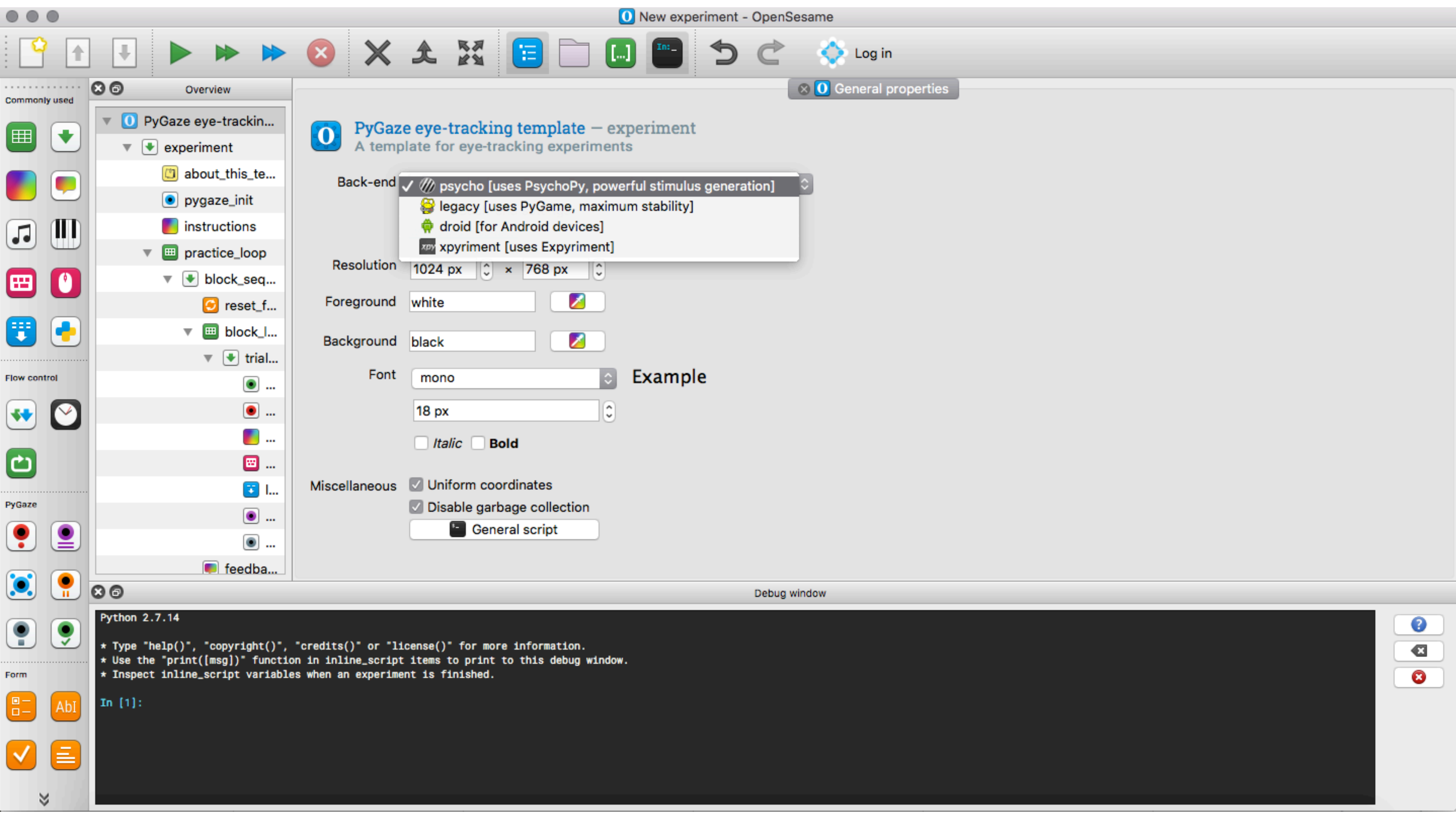
Donate through PayPal

- pygaze_init
- pygaze_start_recording
- pygaze_drift_correct
- pygaze_wait
- pygaze_log
- pygaze_stop_recording

```
'L', 'W', 'R', 'X']
o resume.

The experiment was aborted

item-stack: experiment[run].practice_loop[run].block_sequence[run].block_loop[run].trial_sequence[run].response_T2[run]
user_triggered: True
time: Tue Oct 23 14:36:43 2018

In [1]:
```

Eye-tracking Template

Before we will go through each step of creating an experiment, let’s have a look at the eyetracking functions and how they are placed in the eye tracking template



PyGaze eye-tracking template — experiment
A template for eye-tracking experiments

Back-end: ✓ psycho [uses PsychoPy, powerful stimulus generation]
legacy [uses PyGame, maximum stability]
droid [for Android devices]
xpyriment [uses Expyriment]

Resolution: 1024 px × 768 px

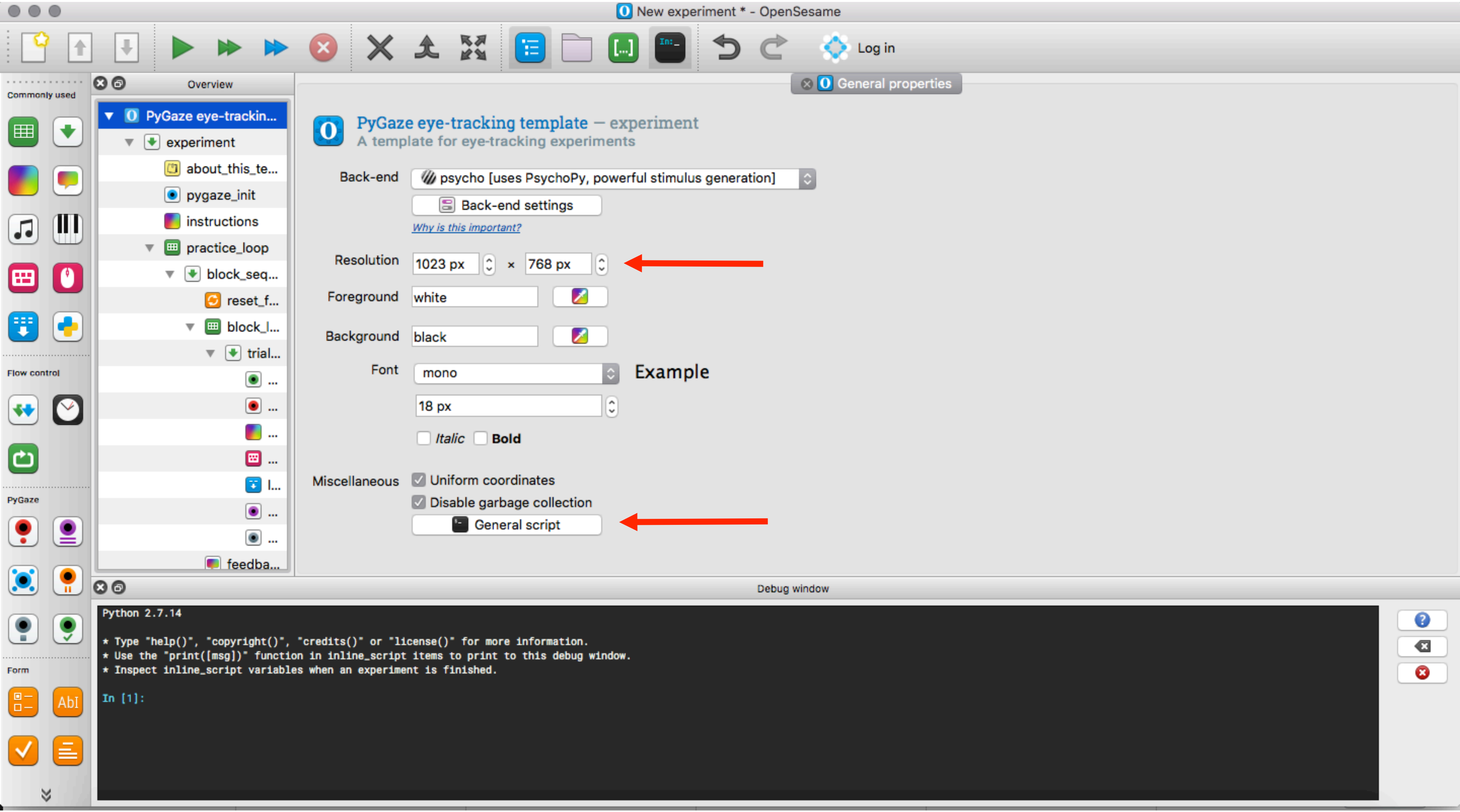
Foreground: white

Background: black

Font: mono, 18 px
□ *Italic* □ **Bold**

Miscellaneous:
☑ Uniform coordinates
☑ Disable garbage collection

```
Python 2.7.14

* Type "help()", "copyright()", "credits()" or "license()" for more information.
* Use the "print([msg])" function in inline_script items to print to this debug window.
* Inspect inline_script variables when an experiment is finished.

In [1]:
```



PyGaze eye-tracking template — experiment
A template for eye-tracking experiments

Back-end: psycho [uses PsychoPy, powerful stimulus generation]

Resolution: 1023 px × 768 px

Foreground: white

Background: black

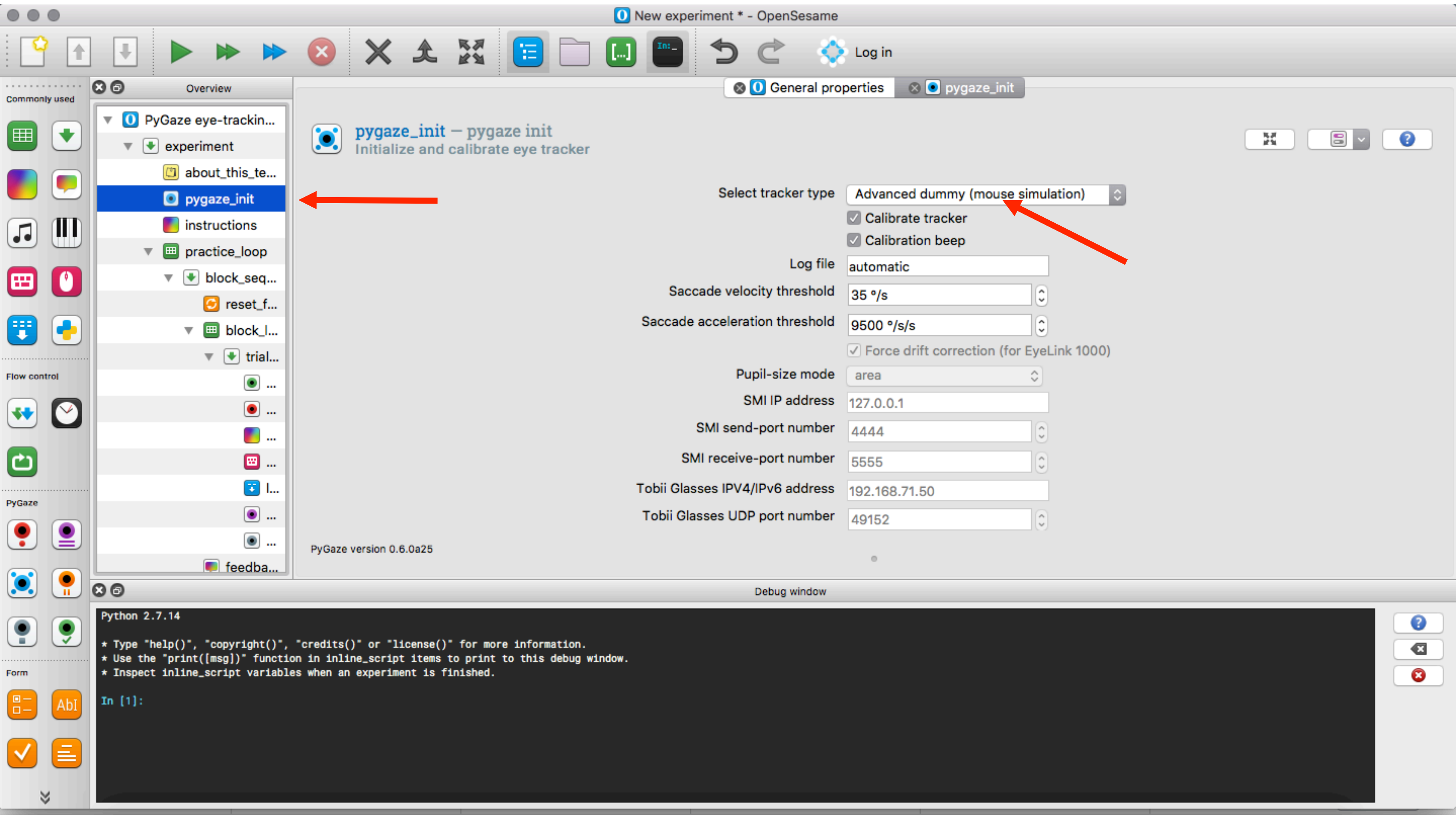
Font: mono

18 px

□ *Italic* □ **Bold**

Miscellaneous: ☑ Uniform coordinates ☑ Disable garbage collection

```
Python 2.7.14

* Type "help()", "copyright()", "credits()" or "license()" for more information.
* Use the "print([msg])" function in inline_script items to print to this debug window.
* Inspect inline_script variables when an experiment is finished.

In [1]:
```



pygaze_init — pygaze init
Initialize and calibrate eye tracker

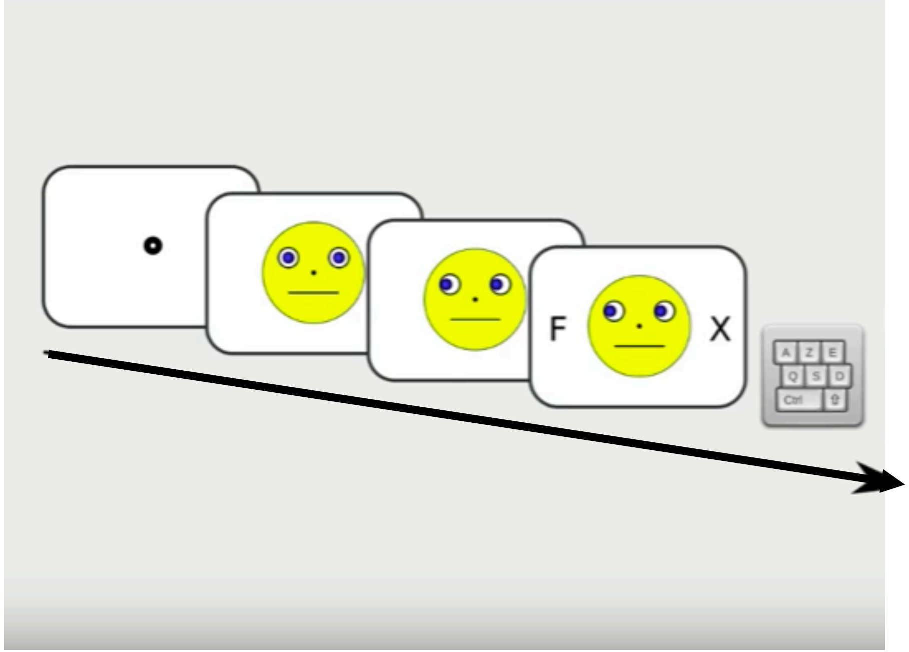
Select tracker type	Advanced dummy (mouse simulation)
	☑ Calibrate tracker
	☑ Calibration beep
Log file	automatic
Saccade velocity threshold	35 °/s
Saccade acceleration threshold	9500 °/s/s
	☑ Force drift correction (for EyeLink 1000)
Pupil-size mode	area
SMI IP address	127.0.0.1
SMI send-port number	4444
SMI receive-port number	5555
Tobii Glasses IPV4/IPv6 address	192.168.71.50
Tobii Glasses UDP port number	49152

PyGaze version 0.6.0a25

```
Python 2.7.14

* Type "help()", "copyright()", "credits()" or "license()" for more information.
* Use the "print([msg])" function in inline_script items to print to this debug window.
* Inspect inline_script variables when an experiment is finished.

In [1]:
```

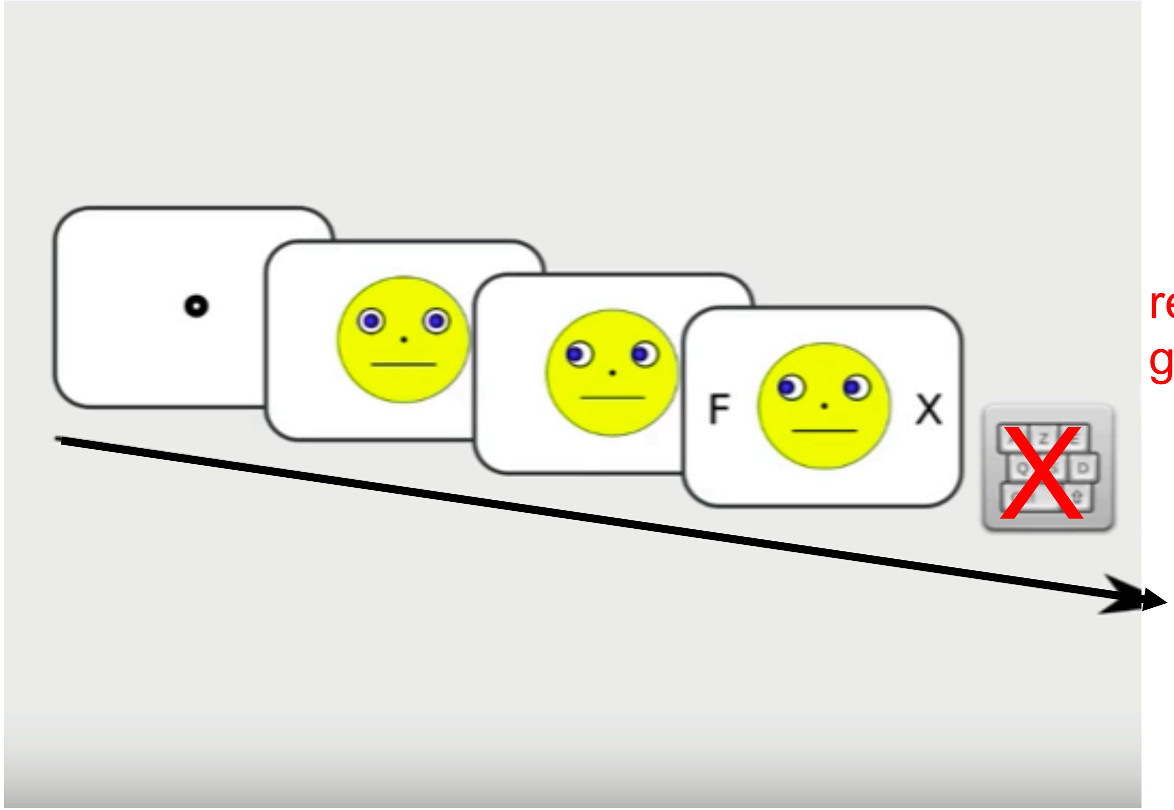
To do list

One “usual” behavioural experiment



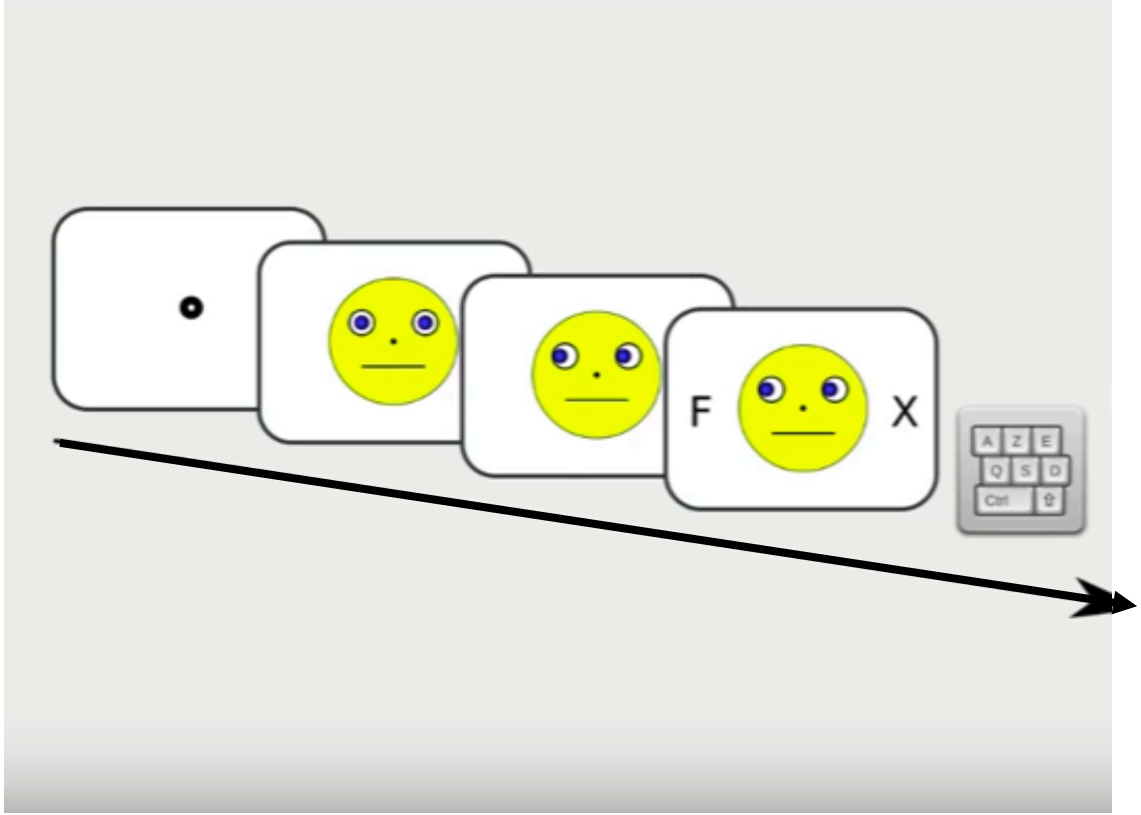
One gaze-contingent experiment 👁

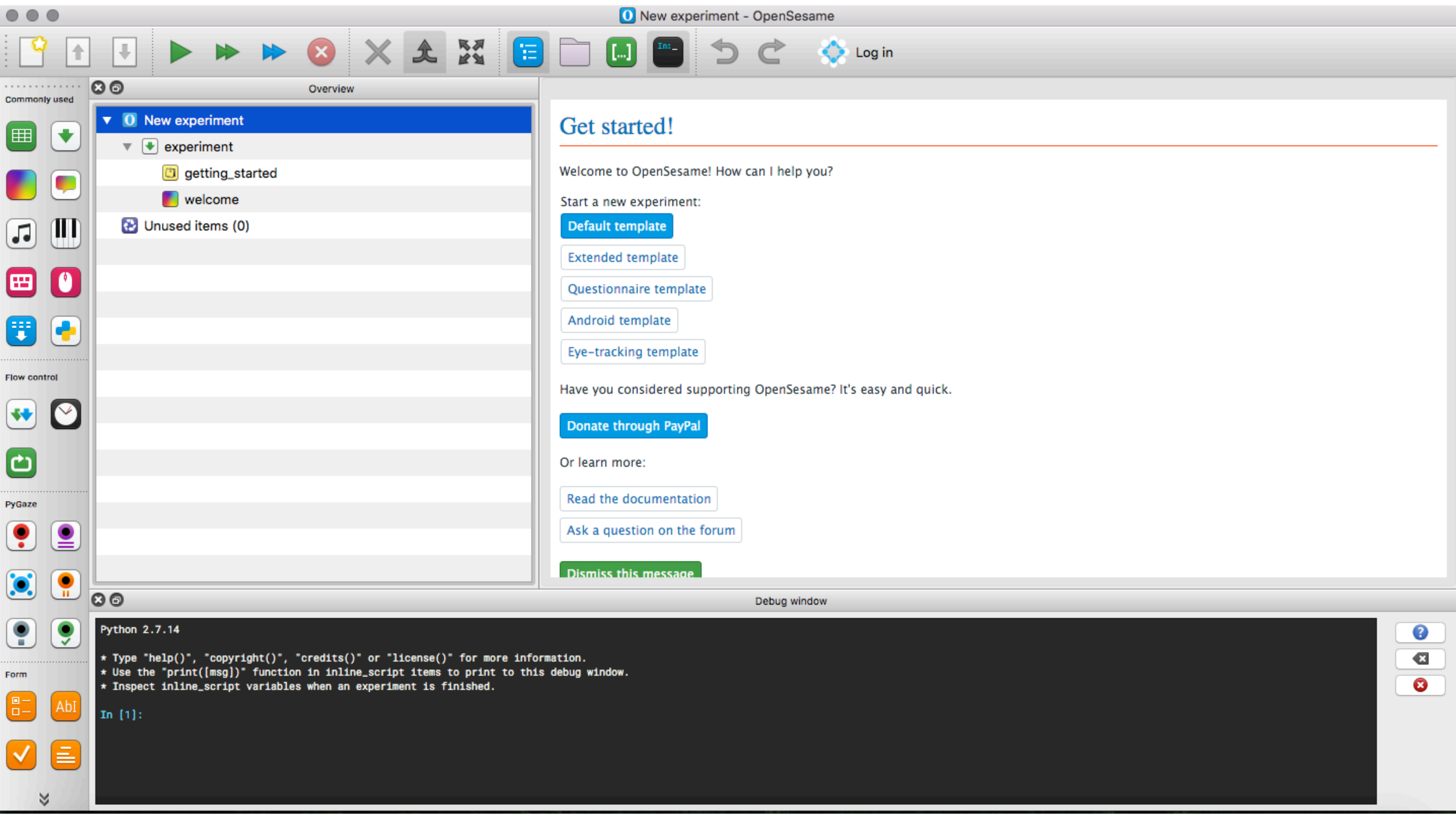
If we have time: combine them in one



replaced with
gaze fixation

1. Simple trail-based experiment





Get started!

Welcome to OpenSesame! How can I help you?

Start a new experiment:

Have you considered supporting OpenSesame? It's easy and quick.

Or learn more:

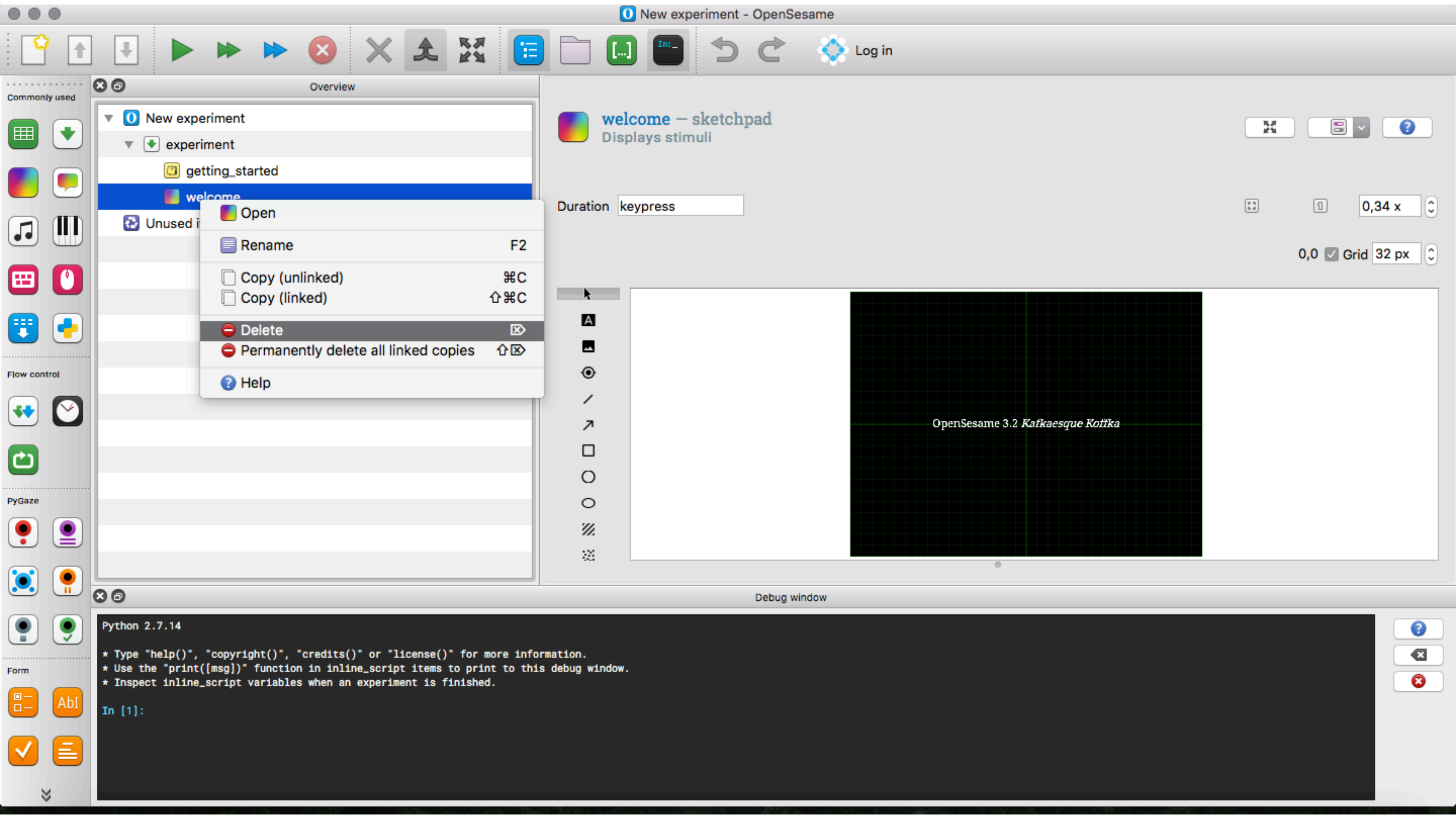
```
Python 2.7.14

* Type "help()", "copyright()", "credits()" or "license()" for more information.
* Use the "print([msg])" function in inline_script items to print to this debug window.
* Inspect inline_script variables when an experiment is finished.

In [1]:
```

OpenSesame paradigm

Think from the highest hierarchy to the smaller elements

The experimental structure

1. Provide the instructions to the participant
2. Main experimental block
- Loop of repeated trials
1. Save data (will return to it later)
2. End screen for the participant

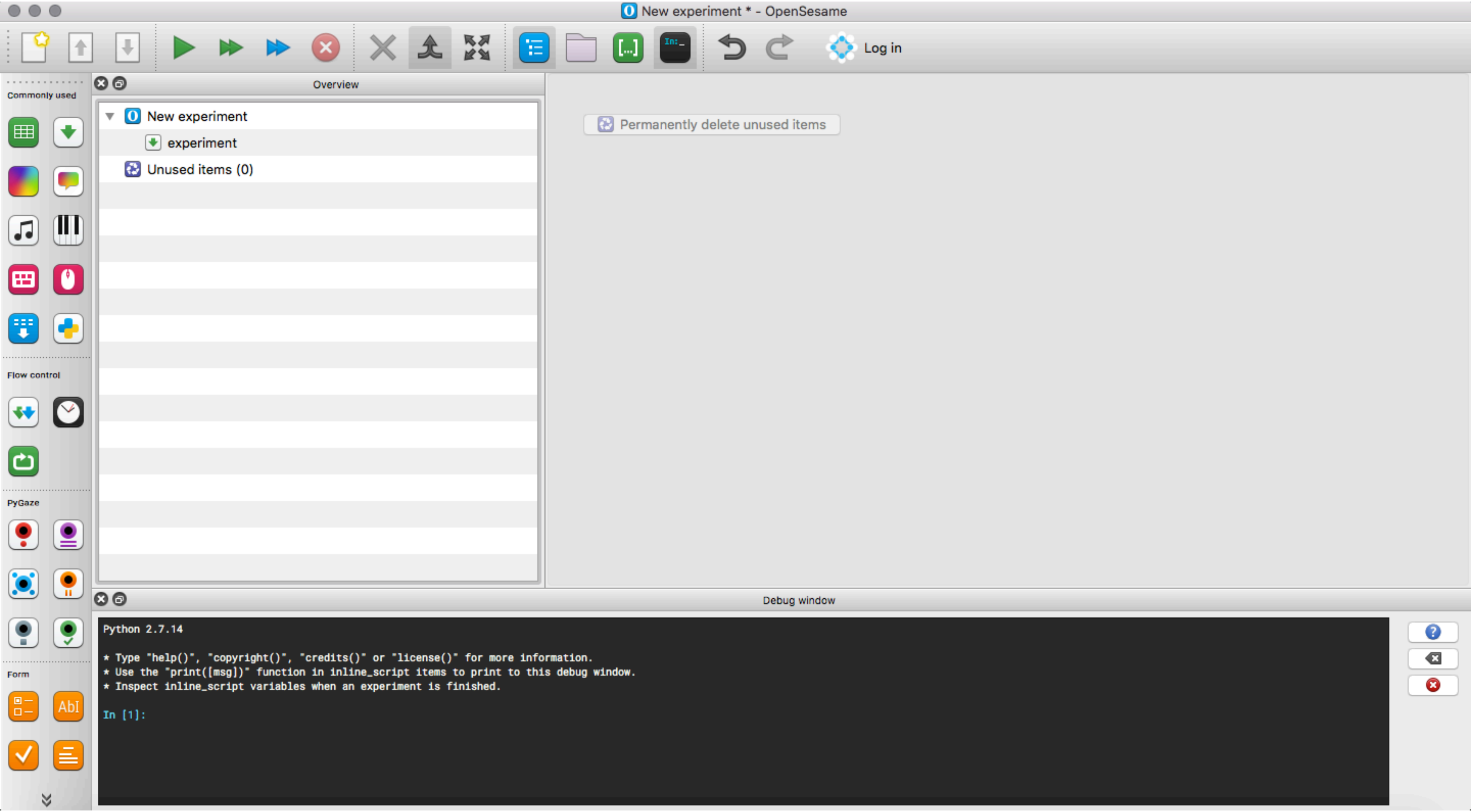


Duration keypress

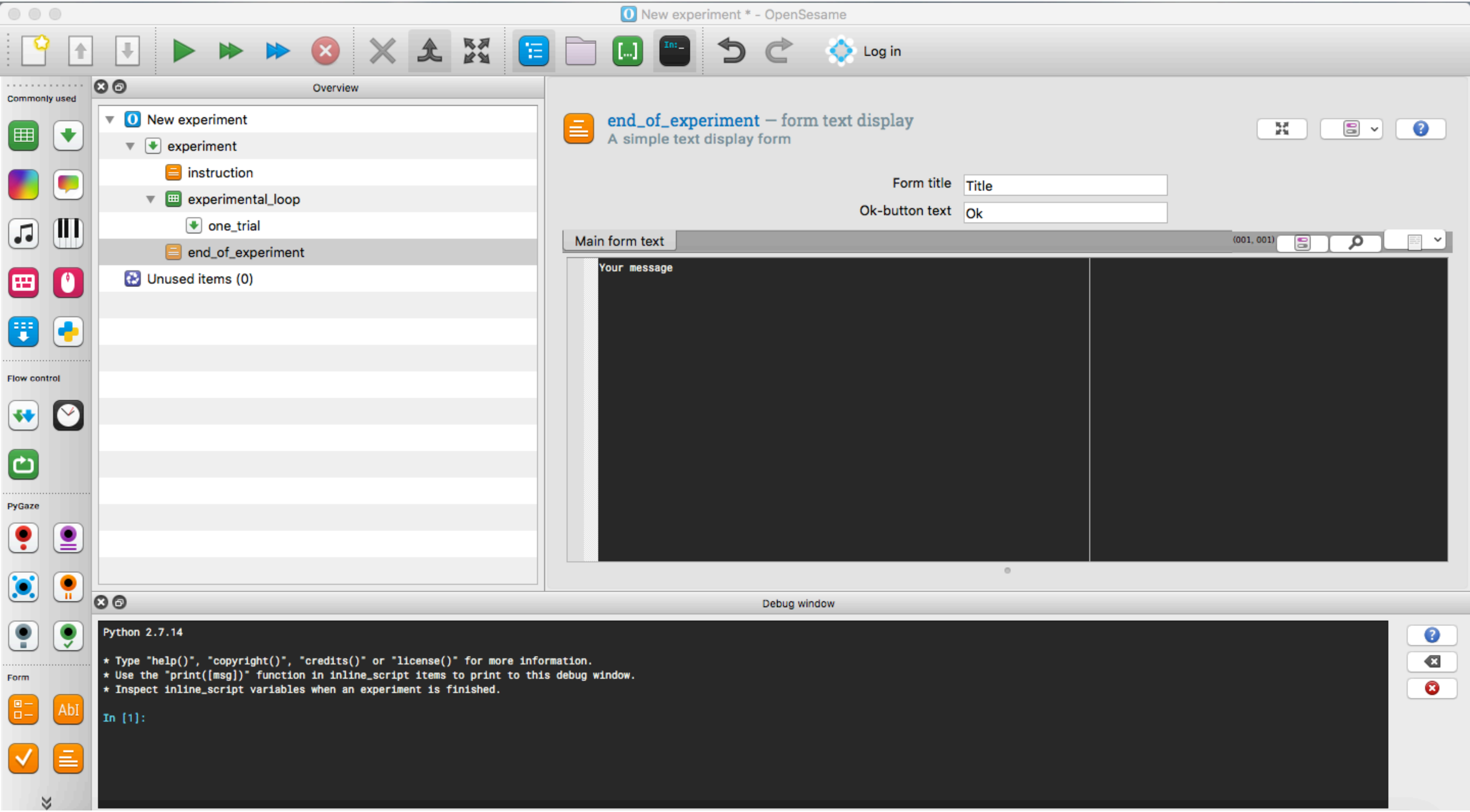
0,34 x

0,0 ☑ Grid 32 px

```
Python 2.7.14

* Type "help()", "copyright()", "credits()" or "license()" for more information.
* Use the "print([msg])" function in inline_script items to print to this debug window.
* Inspect inline_script variables when an experiment is finished.

In [1]:
```



```
Python 2.7.14

* Type "help()", "copyright()", "credits()" or "license()" for more information.
* Use the "print([msg])" function in inline_script items to print to this debug window.
* Inspect inline_script variables when an experiment is finished.

In [1]:
```

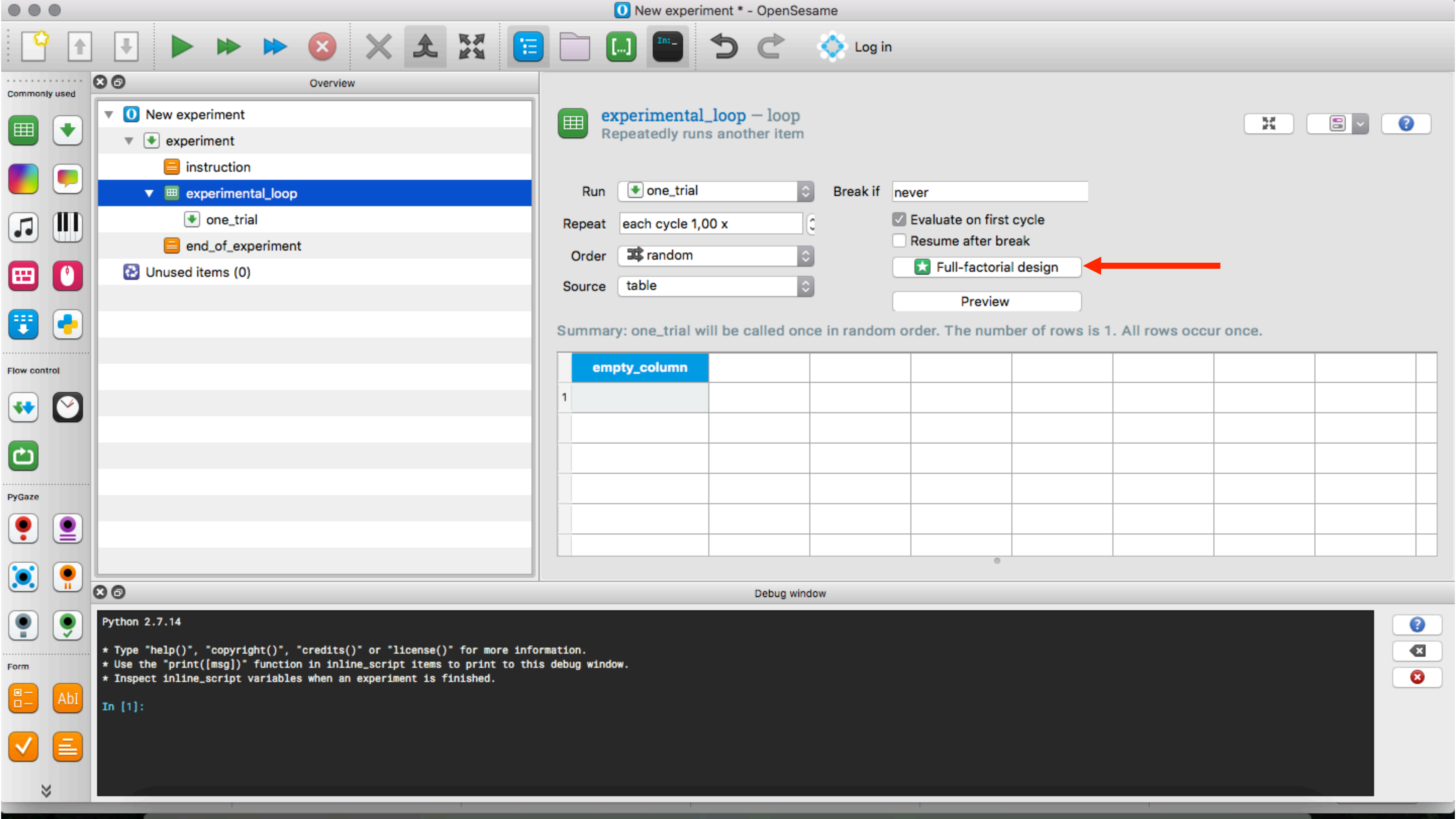


Form title Title

Ok-button text Ok

```
Your message
```

```
Python 2.7.14

* Type "help()", "copyright()", "credits()" or "license()" for more information.
* Use the "print([msg])" function in inline_script items to print to this debug window.
* Inspect inline_script variables when an experiment is finished.

In [1]:
```



experimental_loop — loop
Repeatedly runs another item

Run: one_trial

Repeat: each cycle 1,00 x

Order: random

Source: table

Break if: never

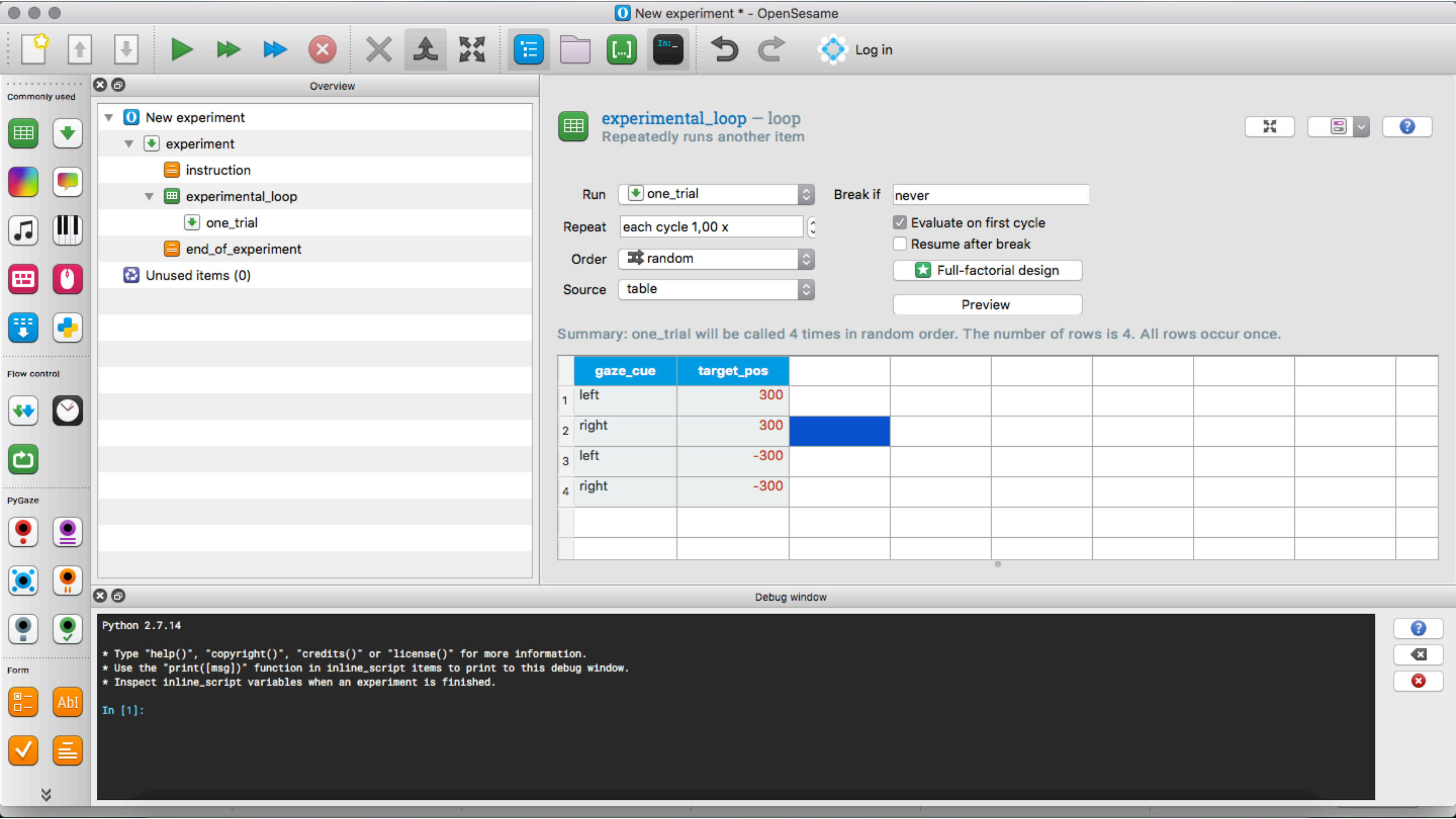
☑ Evaluate on first cycle

□ Resume after break

Summary: one_trial will be called once in random order. The number of rows is 1. All rows occur once.

	empty_column
1	

```
Python 2.7.14

* Type "help()", "copyright()", "credits()" or "license()" for more information.
* Use the "print([msg])" function in inline_script items to print to this debug window.
* Inspect inline_script variables when an experiment is finished.

In [1]:
```



experimental_loop — loop
Repeatedly runs another item

Run: one_trial

Repeat: each cycle 1,00 x

Order: random

Source: table

Break if: never

☑ Evaluate on first cycle

□ Resume after break

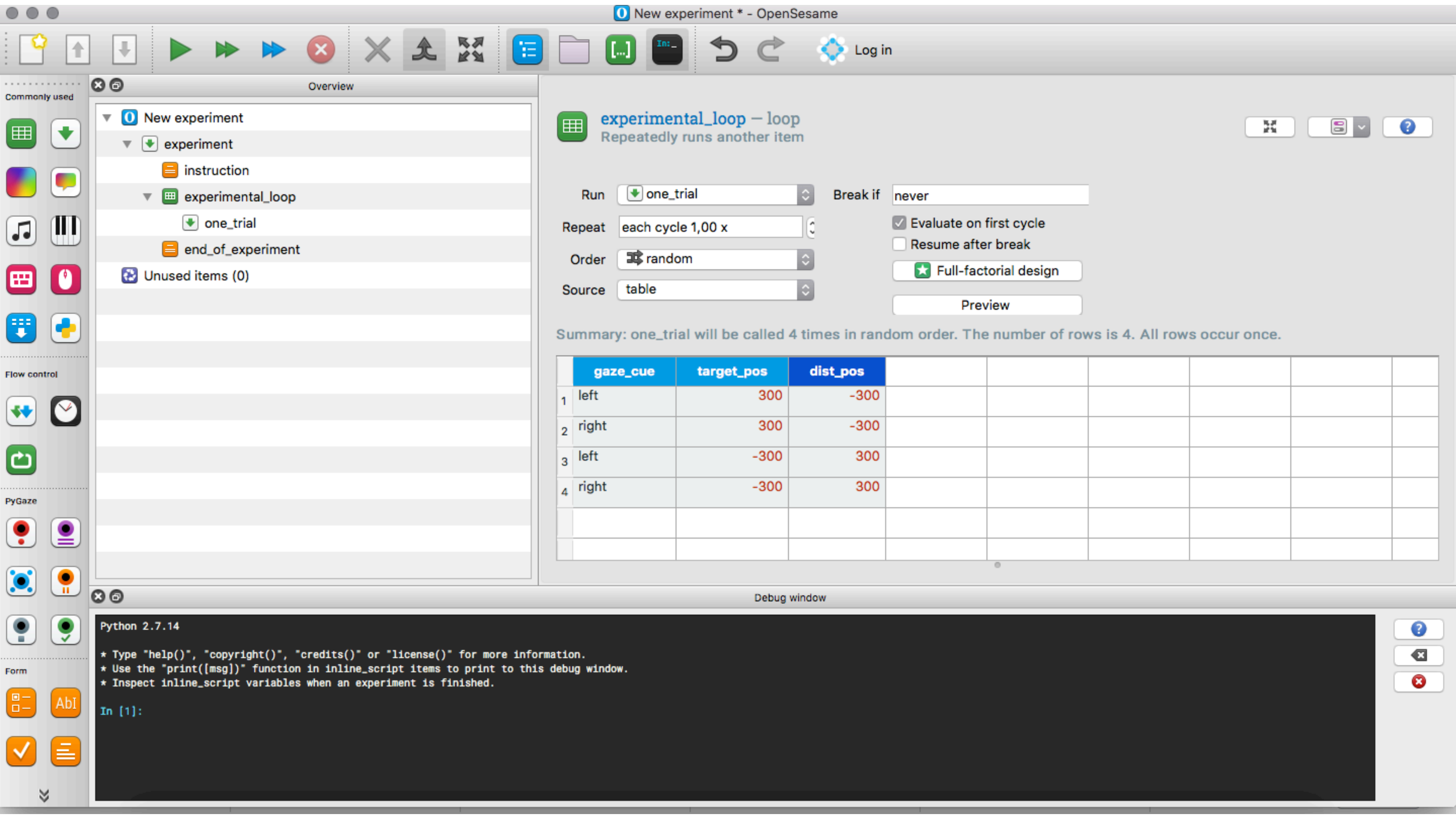
Full-factorial design

Preview

Summary: one_trial will be called 4 times in random order. The number of rows is 4. All rows occur once.

	gaze_cue	target_pos
1	left	300
2	right	300
3	left	-300
4	right	-300

```
Python 2.7.14

* Type "help()", "copyright()", "credits()" or "license()" for more information.
* Use the "print([msg])" function in inline_script items to print to this debug window.
* Inspect inline_script variables when an experiment is finished.

In [1]:
```



experimental_loop – loop
Repeatedly runs another item

Run: one_trial

Repeat: each cycle 1,00 x

Order: random

Source: table

Break if: never

☑ Evaluate on first cycle

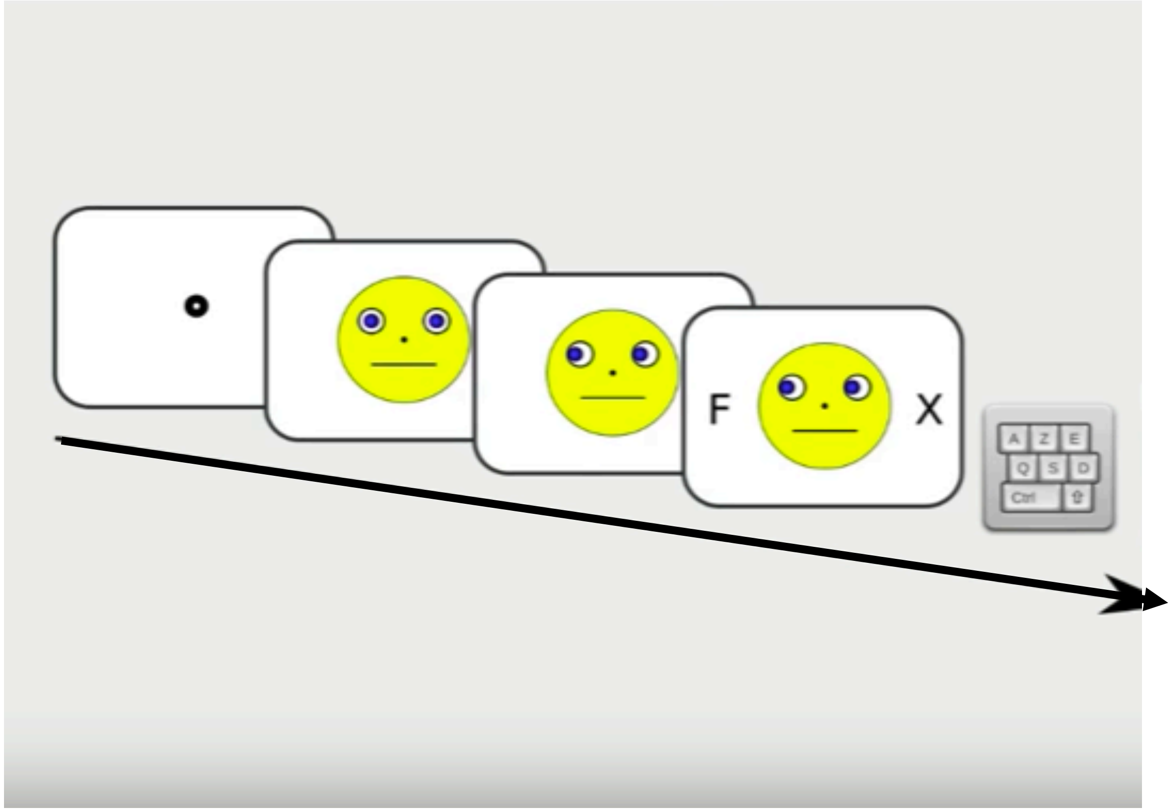
□ Resume after break

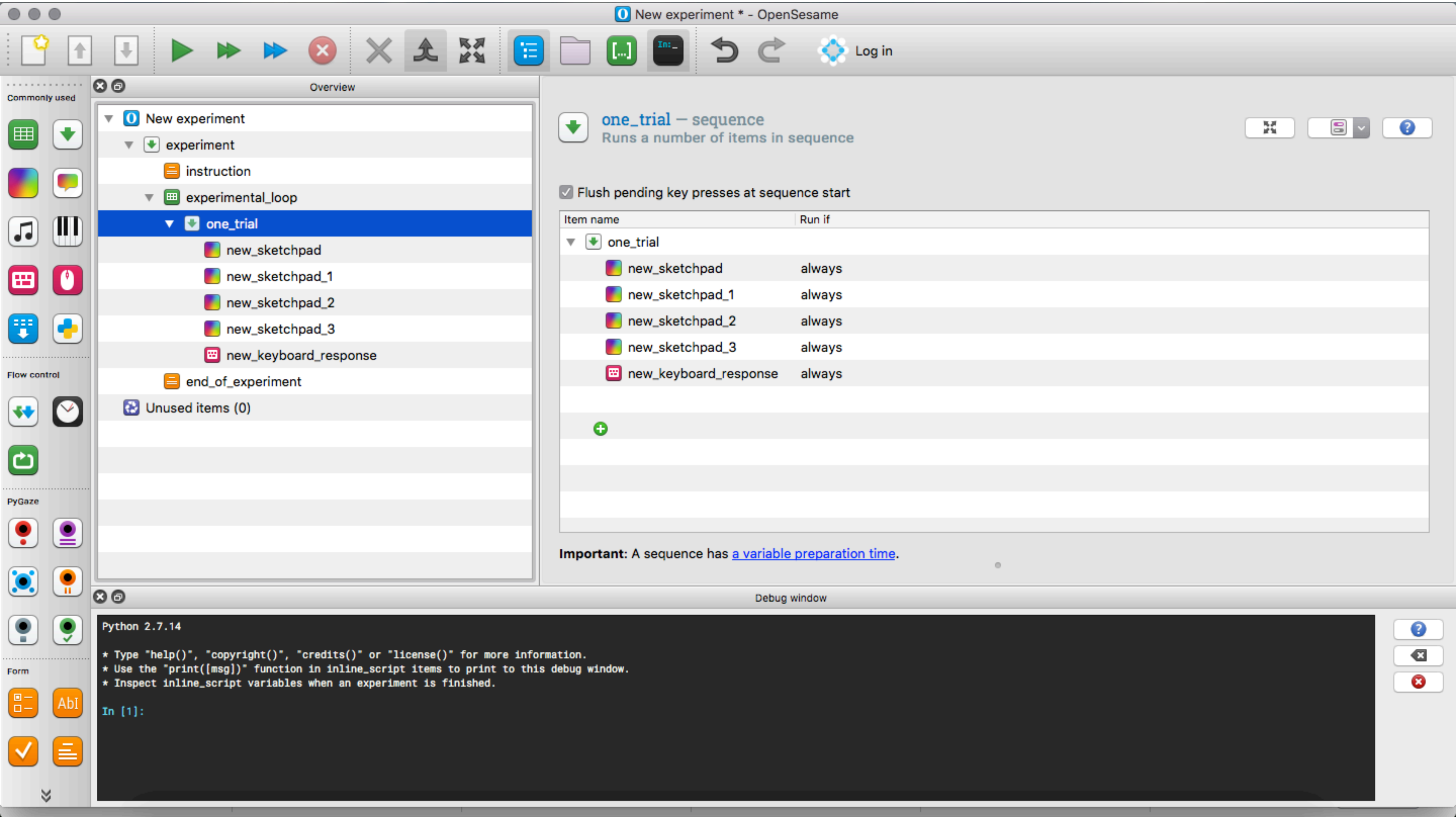
Summary: one_trial will be called 4 times in random order. The number of rows is 4. All rows occur once.

	gaze_cue	target_pos	dist_pos
1	left	300	-300
2	right	300	-300
3	left	-300	300
4	right	-300	300

```
Python 2.7.14

* Type "help()", "copyright()", "credits()" or "license()" for more information.
* Use the "print([msg])" function in inline_script items to print to this debug window.
* Inspect inline_script variables when an experiment is finished.

In [1]:
```

One trial:





- New experiment
  - experiment
    - instruction
    - experimental_loop
      - one_trial
        - new_sketchpad
        - new_sketchpad_1
        - new_sketchpad_2
        - new_sketchpad_3
        - new_keyboard_response
    - end_of_experiment
- Unused items (0)

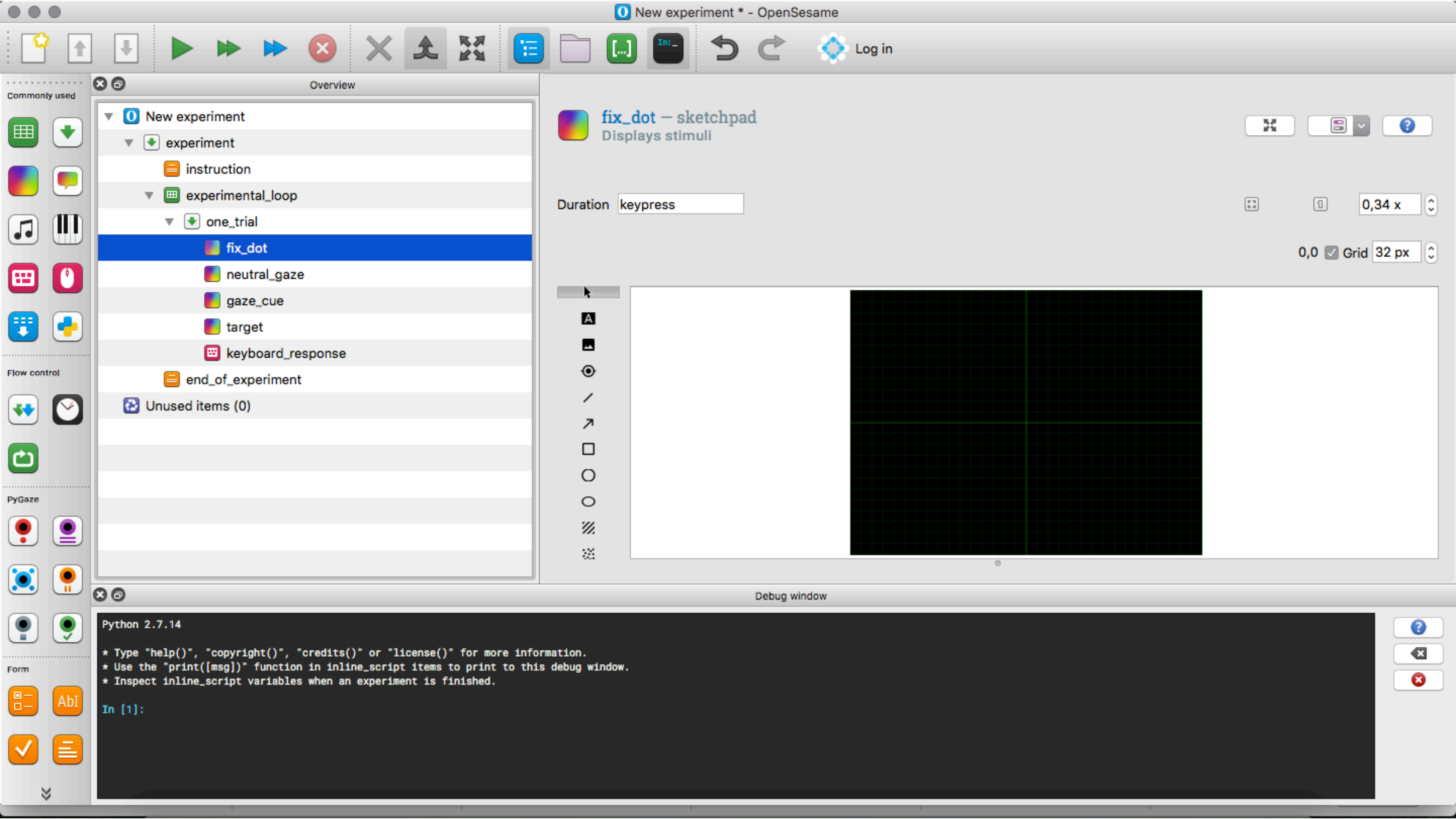
one_trial — sequence
Runs a number of items in sequence

☑ Flush pending key presses at sequence start

Item name	Run if
one_trial	
new_sketchpad	always
new_sketchpad_1	always
new_sketchpad_2	always
new_sketchpad_3	always
new_keyboard_response	always

Important: A sequence has a variable preparation time.

```
Python 2.7.14

* Type "help()", "copyright()", "credits()" or "license()" for more information.
* Use the "print([msg])" function in inline_script items to print to this debug window.
* Inspect inline_script variables when an experiment is finished.

In [1]:
```

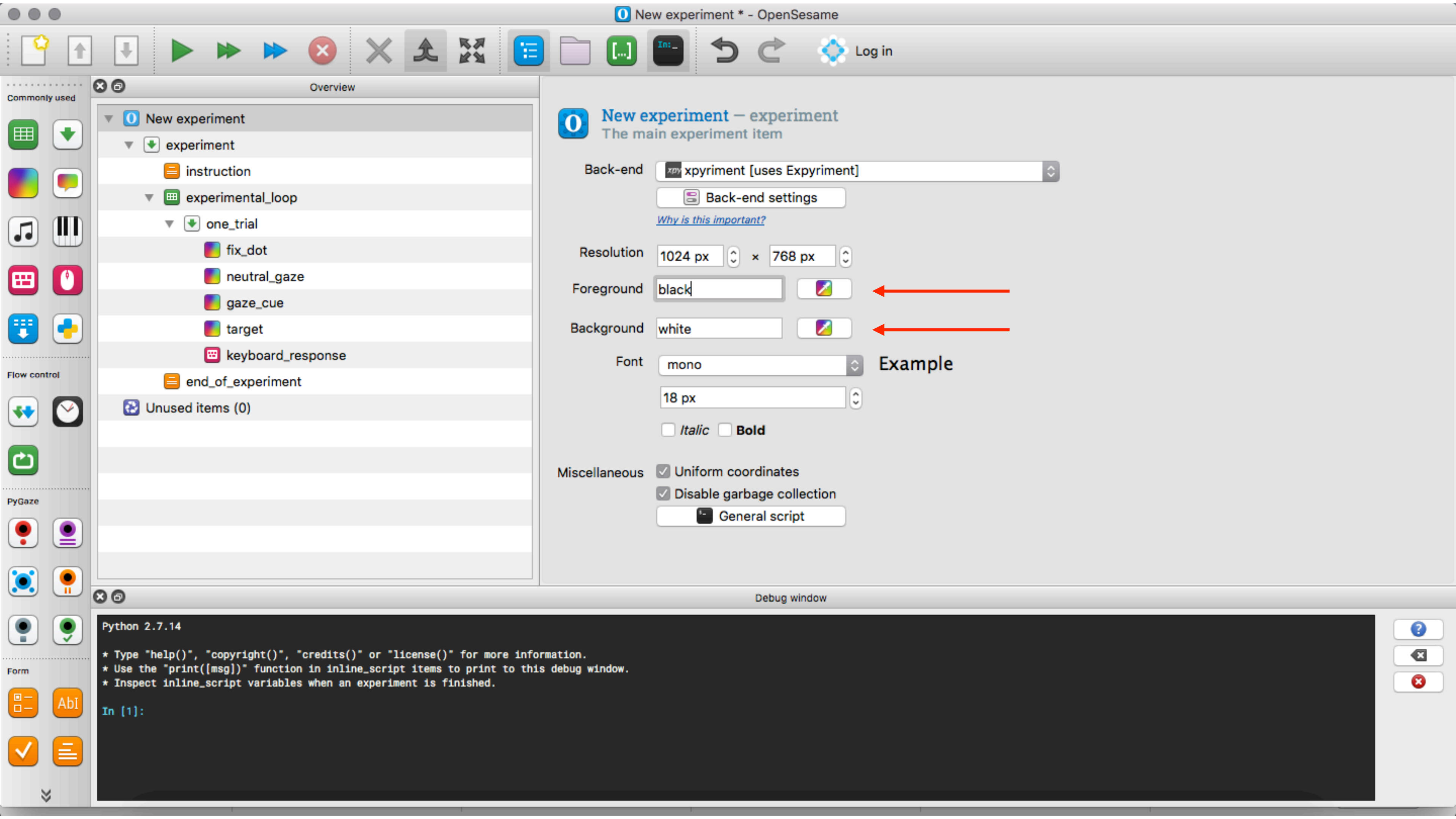


Duration keypress

0,34 x

Grid 32 px

```
Python 2.7.14

* Type "help()", "copyright()", "credits()" or "license()" for more information.
* Use the "print([msg])" function in inline_script items to print to this debug window.
* Inspect inline_script variables when an experiment is finished.

In [1]:
```



New experiment — experiment
The main experiment item

Back-end: xpyriment [uses Expyriment]

Resolution: 1024 px × 768 px

Foreground: black

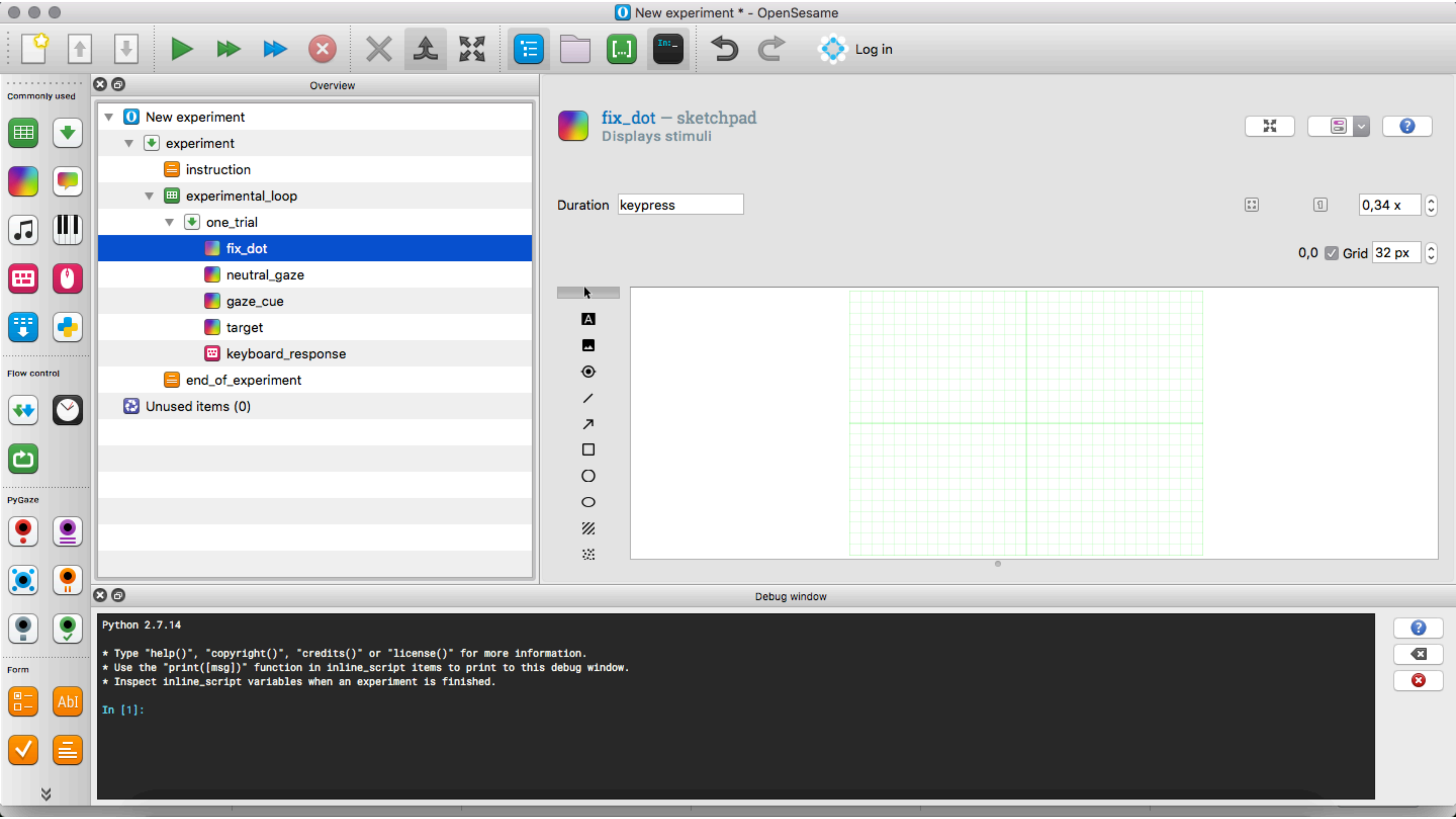
Background: white

Font: mono, 18 px

□ Italic □ Bold

Miscellaneous: ☑ Uniform coordinates ☑ Disable garbage collection

```
Python 2.7.14

* Type "help()", "copyright()", "credits()" or "license()" for more information.
* Use the "print([msg])" function in inline_script items to print to this debug window.
* Inspect inline_script variables when an experiment is finished.

In [1]:
```



- New experiment
  - experiment
    - instruction
    - experimental_loop
      - one_trial
        - fix_dot
        - neutral_gaze
        - gaze_cue
        - target
        - keyboard_response
    - end_of_experiment
- Unused items (0)

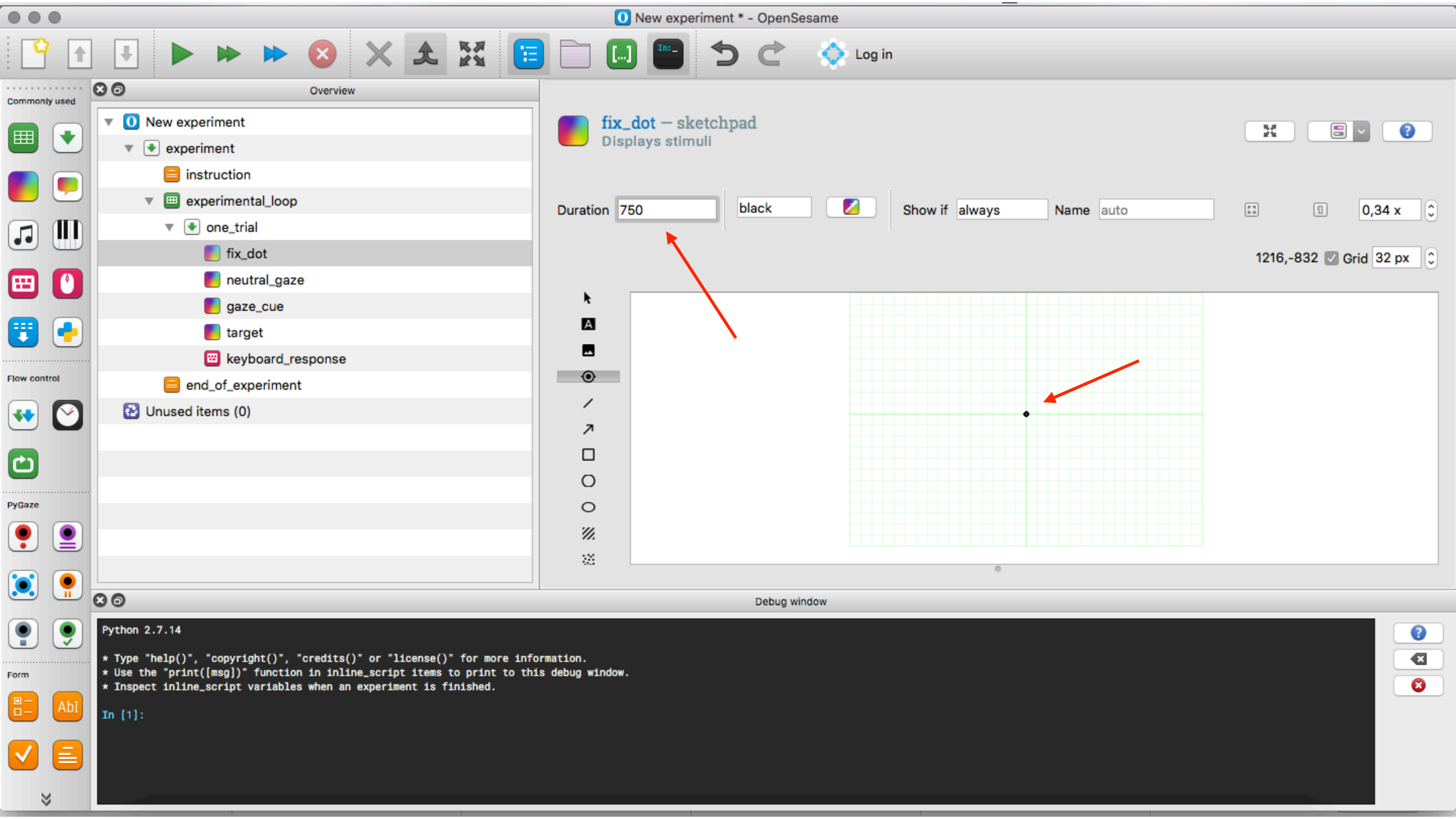
fix_dot – sketchpad
Displays stimuli

Duration keypress

0,34 x

0,0 ☑ Grid 32 px

```
Python 2.7.14

* Type "help()", "copyright()", "credits()" or "license()" for more information.
* Use the "print([msg])" function in inline_script items to print to this debug window.
* Inspect inline_script variables when an experiment is finished.

In [1]:
```



Duration 750 black

Show if always Name auto 0,34 x

1216,-832 ☑ Grid 32 px

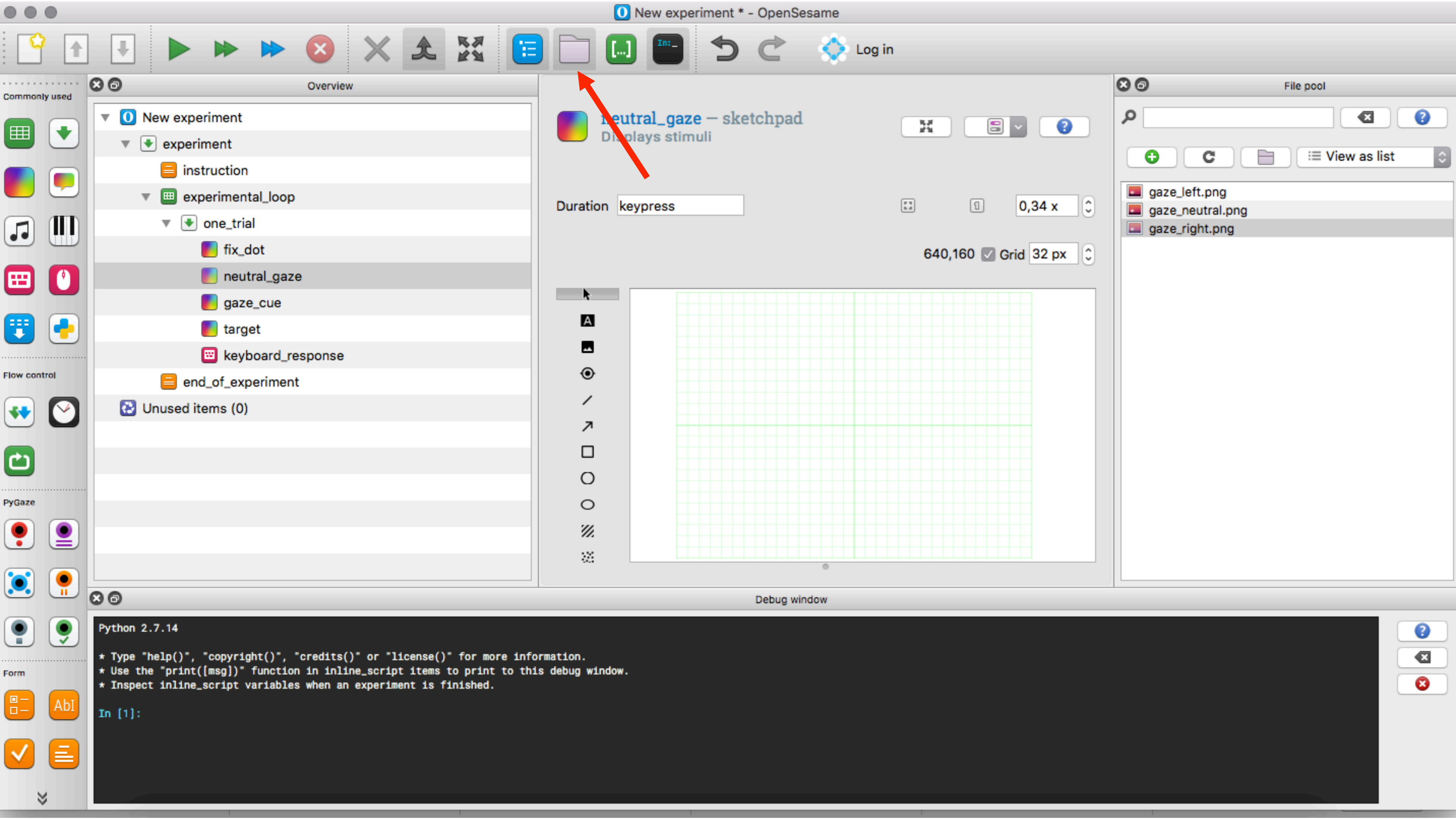
```
Python 2.7.14

* Type "help()", "copyright()", "credits()" or "license()" for more information.
* Use the "print([msg])" function in inline_script items to print to this debug window.
* Inspect inline_script variables when an experiment is finished.

In [1]:
```

Face images

https://osdoc.cogsci.nl/3.2/tutorials/beginner/#step-4-add-images-and-sound-files-to-the-file-pool

Take the pictures from: OpenSesame website -> Beginners Tutorial -> Step 4

Save at any place on your laptop. Don’t change the names

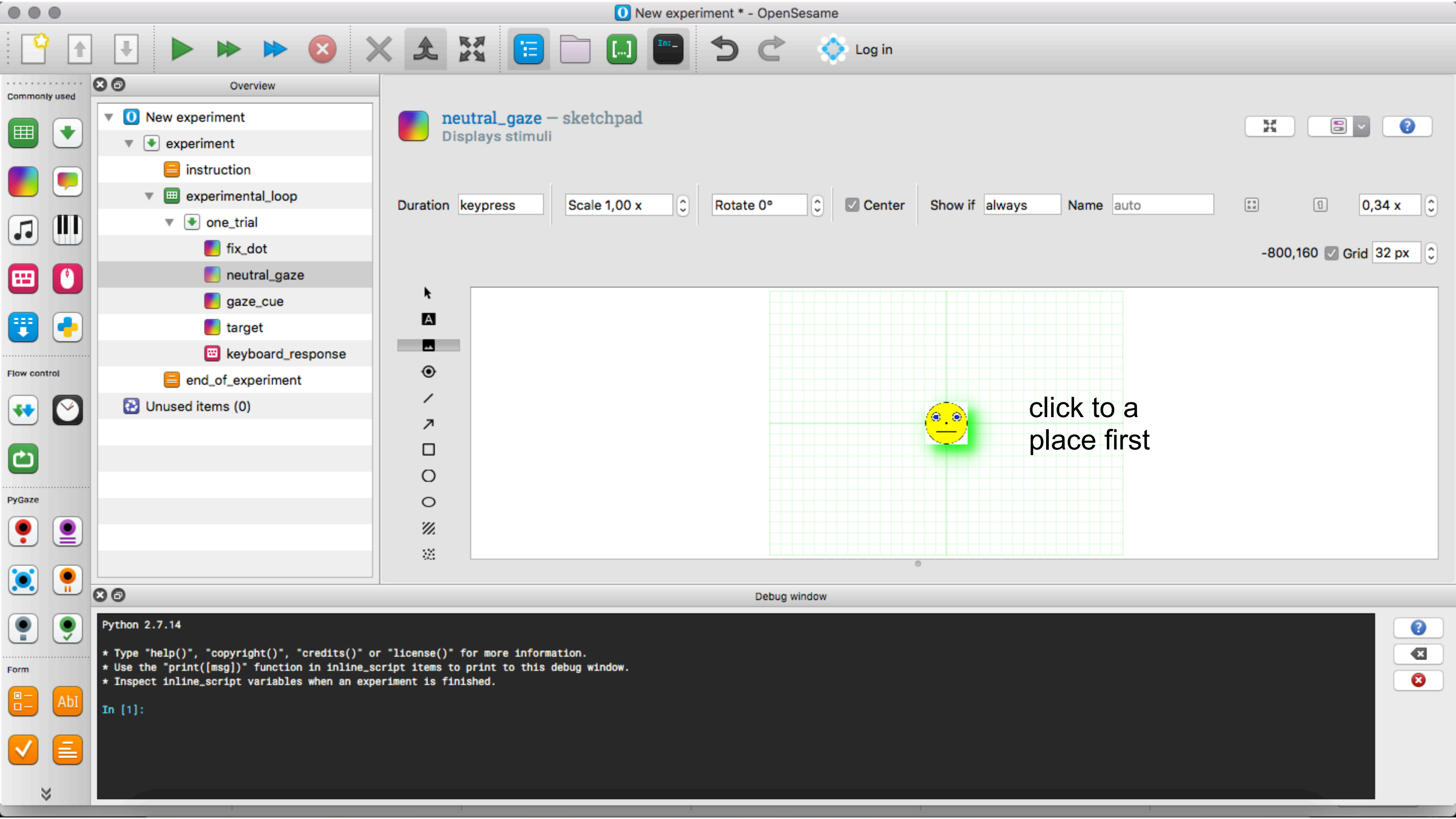


Duration keypress

0,34 x

☑ Grid 32 px

```
Python 2.7.14

* Type "help()", "copyright()", "credits()" or "license()" for more information.
* Use the "print([msg])" function in inline_script items to print to this debug window.
* Inspect inline_script variables when an experiment is finished.

In [1]:
```



- New experiment
  - experiment
    - instruction
    - experimental_loop
      - one_trial
        - fix_dot
        - neutral_gaze
        - gaze_cue
        - target
        - keyboard_response
    - end_of_experiment
- Unused items (0)

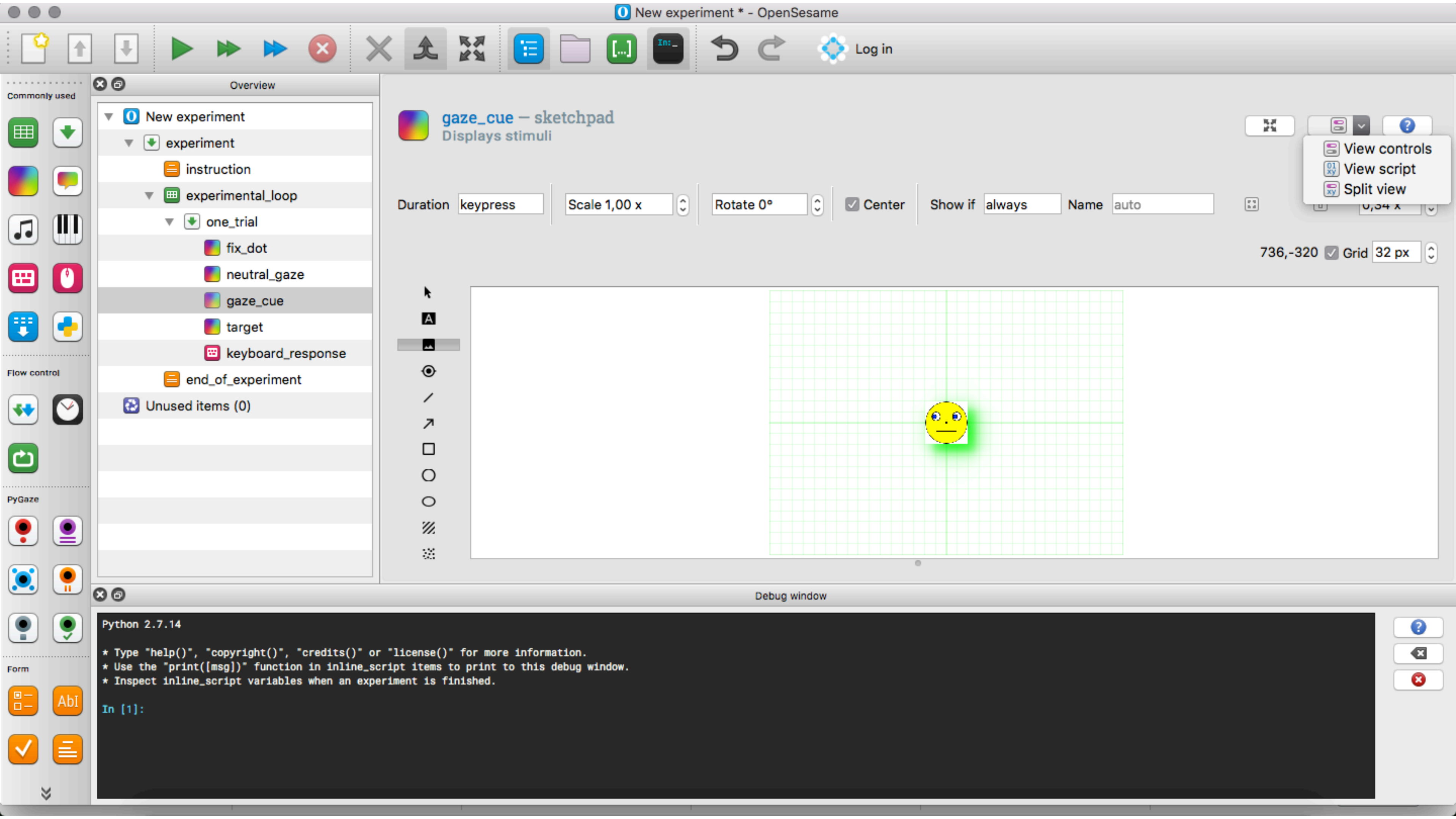
neutral_gaze — sketchpad
Displays stimuli

Duration keypress | Scale 1,00 x | Rotate 0° | ☑ Center | Show if always | Name auto | 0,34 x

-800,160 ☑ Grid 32 px

click to a
place first

```
Python 2.7.14

* Type "help()", "copyright()", "credits()" or "license()" for more information.
* Use the "print([msg])" function in inline_script items to print to this debug window.
* Inspect inline_script variables when an experiment is finished.

In [1]:
```



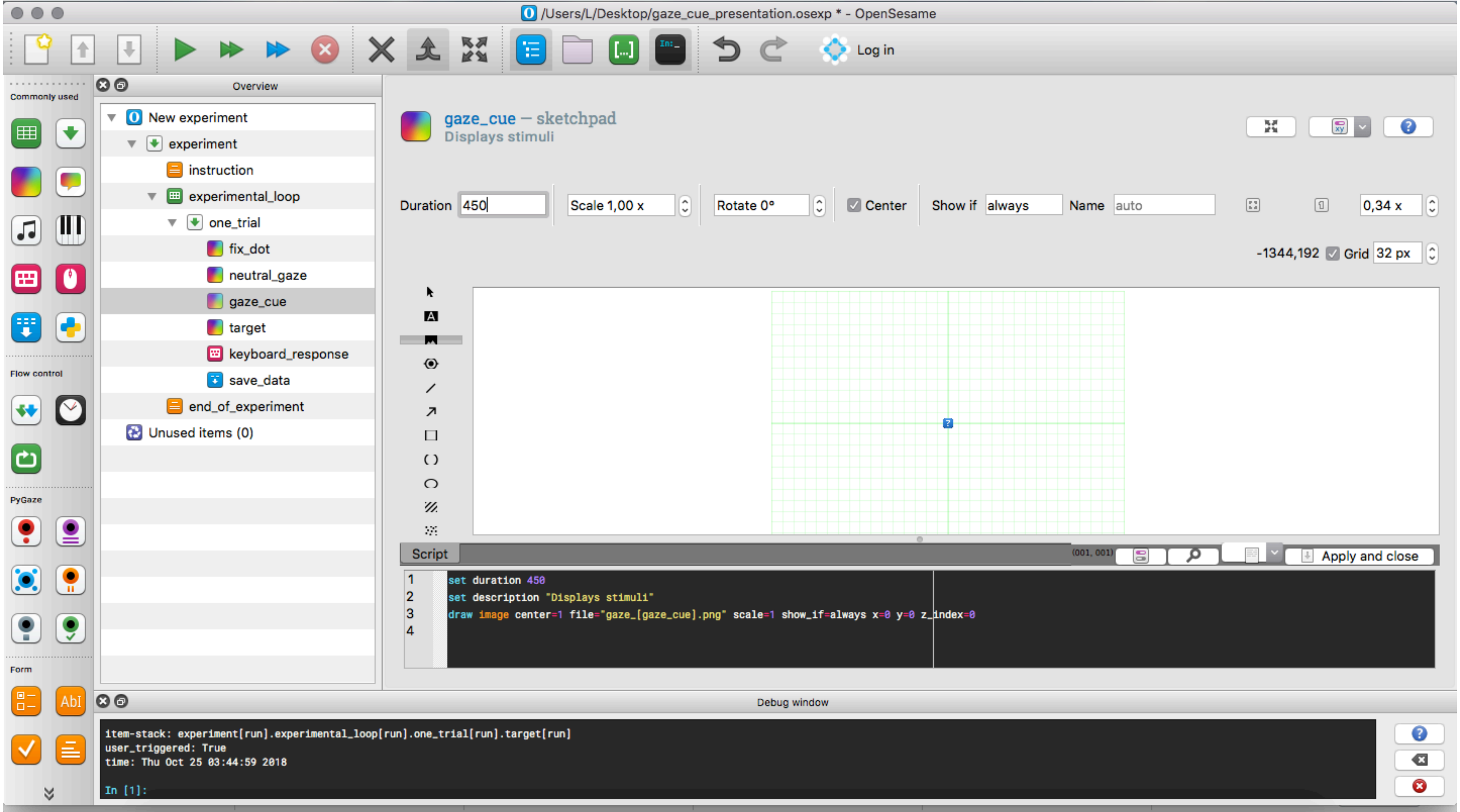
New experiment
- experiment
  - instruction
  - experimental_loop
    - one_trial
      - fix_dot
      - neutral_gaze
      - gaze_cue
      - target
      - keyboard_response
  - end_of_experiment
- Unused items (0)

gaze_cue — sketchpad
Displays stimuli

Duration keypress | Scale 1,00 x | Rotate 0° | ☑ Center | Show if always | Name auto

736,-320 ☑ Grid 32 px

```
Python 2.7.14

* Type "help()", "copyright()", "credits()" or "license()" for more information.
* Use the "print([msg])" function in inline_script items to print to this debug window.
* Inspect inline_script variables when an experiment is finished.

In [1]:
```



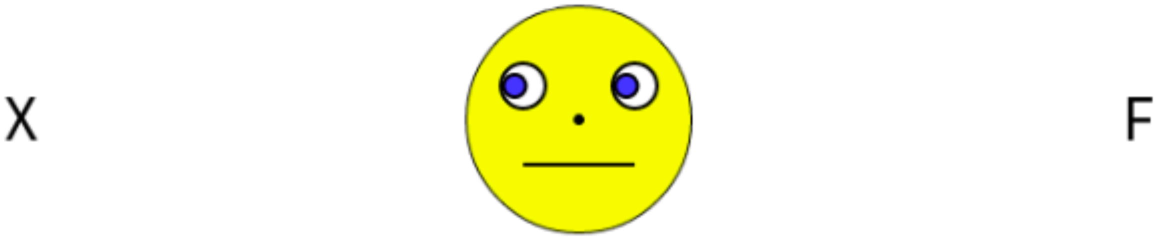
Duration 450 | Scale 1,00 x | Rotate 0° | ☑ Center | Show if always | Name auto | 0,34 x

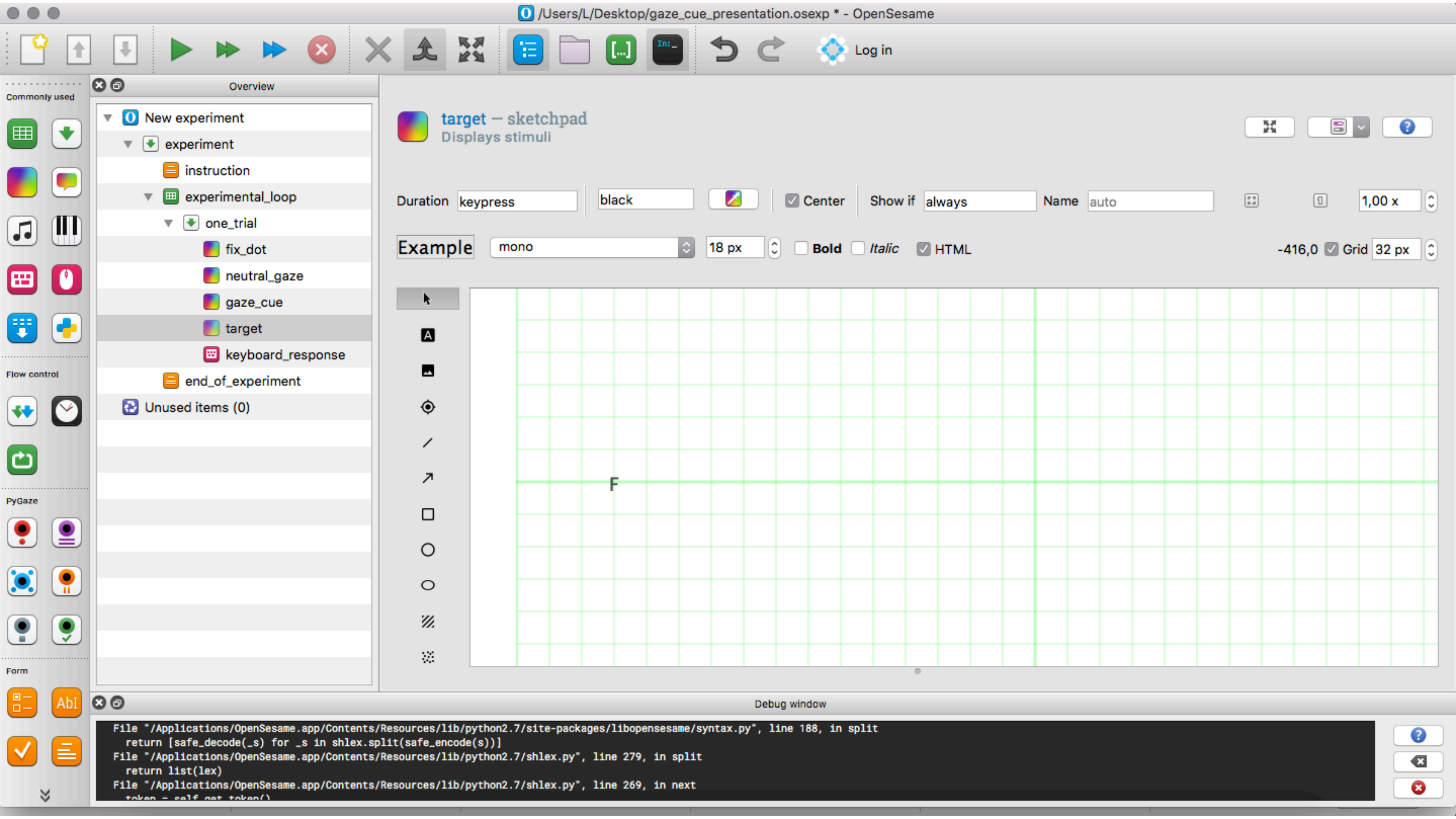
-1344,192 ☑ Grid 32 px

```
set duration 450
set description "Displays stimuli"
draw image center=1 file="gaze_[gaze_cue].png" scale=1 show_if=always x=0 y=0 z_index=0
```

```
item-stack: experiment[run].experimental_loop[run].one_trial[run].target[run]
user_triggered: True
time: Thu Oct 25 03:44:59 2018

In [1]:
```

Do the same for the “target” sketchpad





/Users/L/Desktop/gaze_cue_presentation.osexp * - OpenSesame

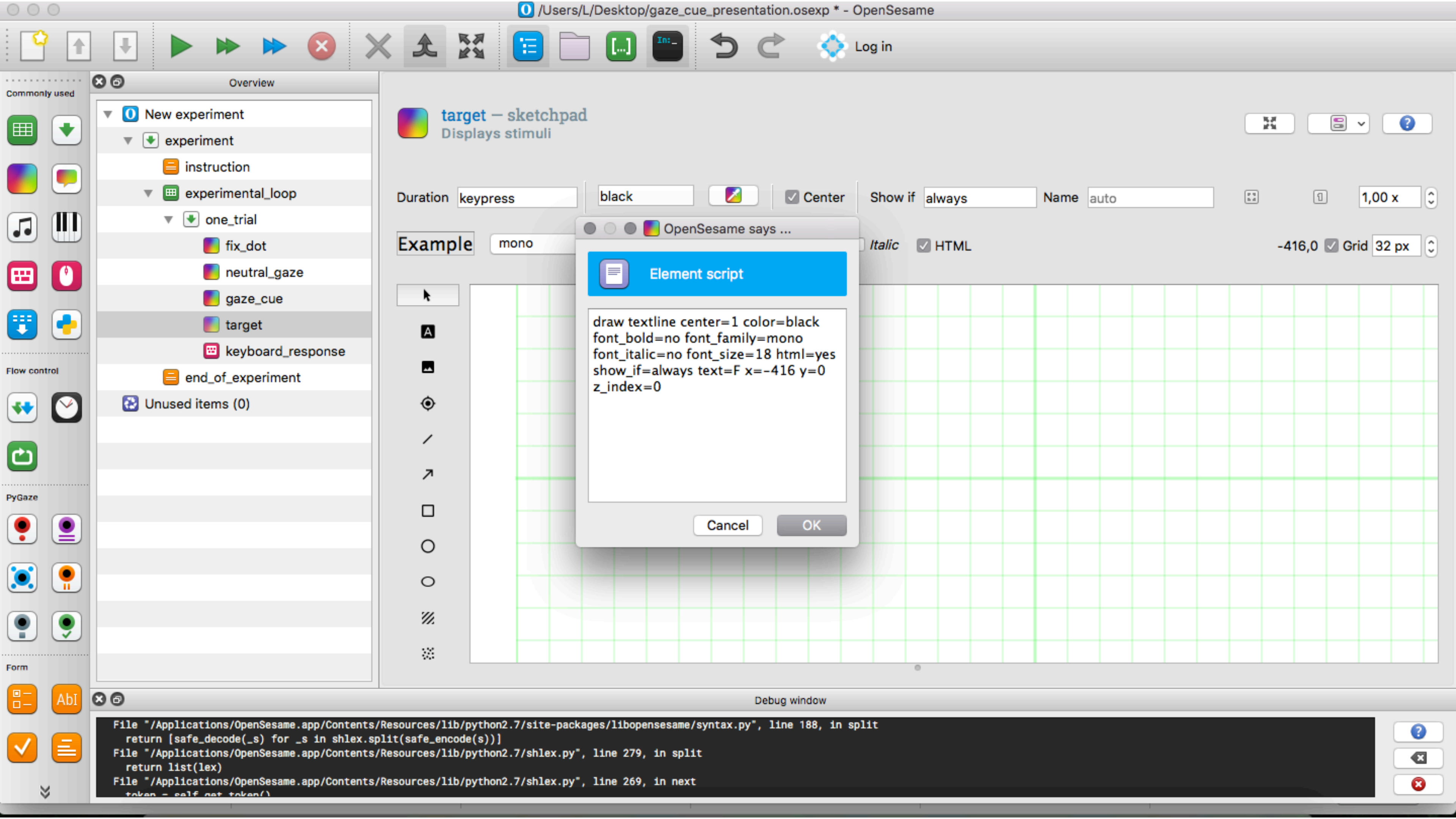
- New experiment
  - experiment
    - instruction
    - experimental_loop
      - one_trial
        - fix_dot
        - neutral_gaze
        - gaze_cue
        - target
        - keyboard_response
    - end_of_experiment
- Unused items (0)

target — sketchpad
Displays stimuli

Duration keypress | black | ☑ Center | Show if always | Name auto | 1,00 x

mono | 18 px | □ Bold | □ *Italic* | ☑ HTML | -416,0 | ☑ Grid 32 px

```
  File "/Applications/OpenSesame.app/Contents/Resources/lib/python2.7/site-packages/libopensesame/syntax.py", line 188, in split
    return [safe_decode(_s) for _s in shlex.split(safe_encode(s))]
  File "/Applications/OpenSesame.app/Contents/Resources/lib/python2.7/shlex.py", line 279, in split
    return list(lex)
  File "/Applications/OpenSesame.app/Contents/Resources/lib/python2.7/shlex.py", line 269, in next
    token = self.get_token()
```



/Users/L/Desktop/gaze_cue_presentation.osexp * - OpenSesame

target — sketchpad
Displays stimuli

Duration keypress | black | ☑ Center | Show if always | Name auto | 1,00 x

Example | mono | Italic | ☑ HTML | -416,0 | ☑ Grid 32 px

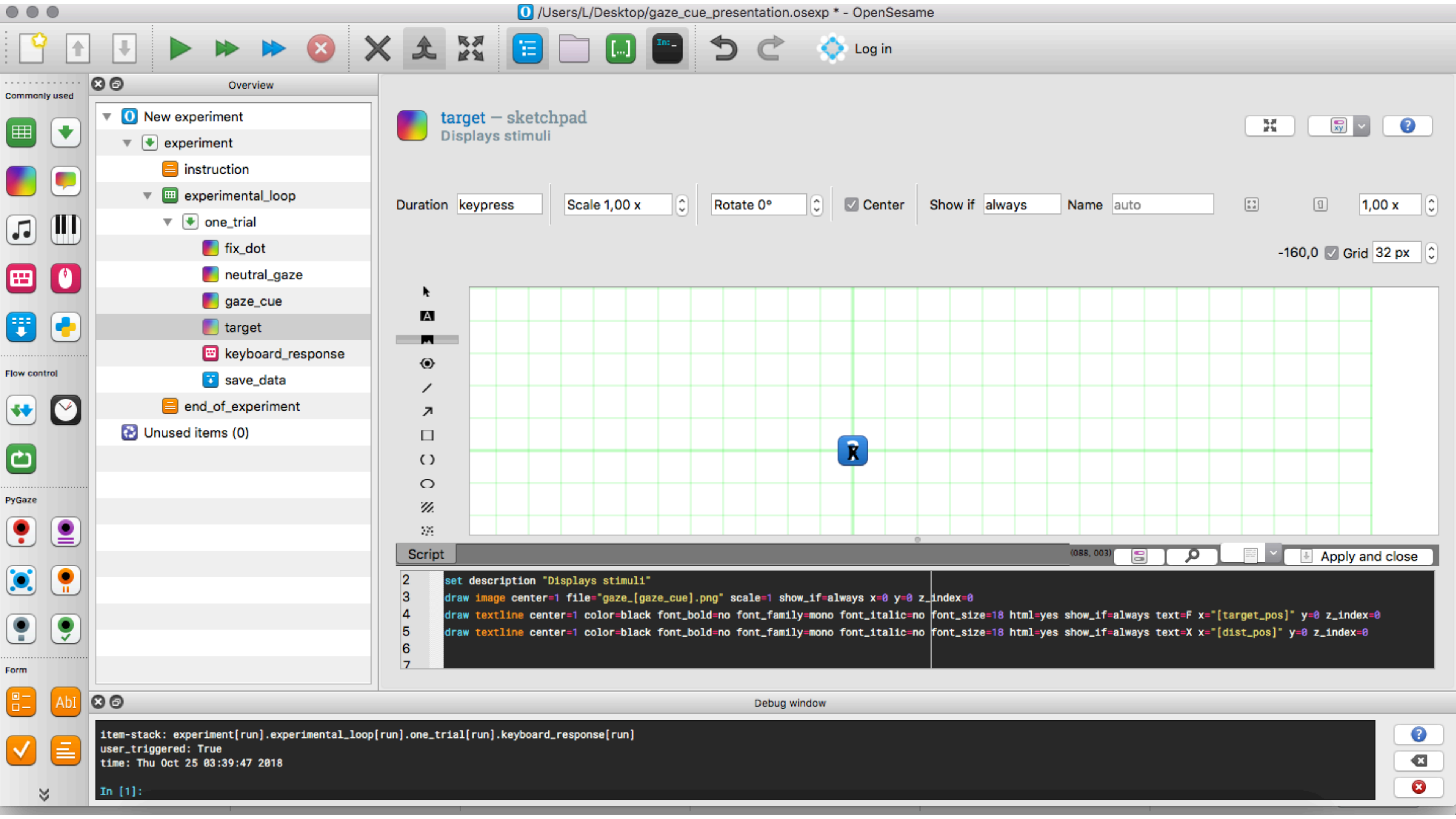
OpenSesame says ...

Element script

```
draw textline center=1 color=black
font_bold=no font_family=mono
font_italic=no font_size=18 html=yes
show_if=always text=F x=-416 y=0
z_index=0
```

Cancel | OK

```
  File "/Applications/OpenSesame.app/Contents/Resources/lib/python2.7/site-packages/libopensesame/syntax.py", line 188, in split
    return [safe_decode(_s) for _s in shlex.split(safe_encode(s))]
  File "/Applications/OpenSesame.app/Contents/Resources/lib/python2.7/shlex.py", line 279, in split
    return list(lex)
  File "/Applications/OpenSesame.app/Contents/Resources/lib/python2.7/shlex.py", line 269, in next
    token = self.get_token()
```



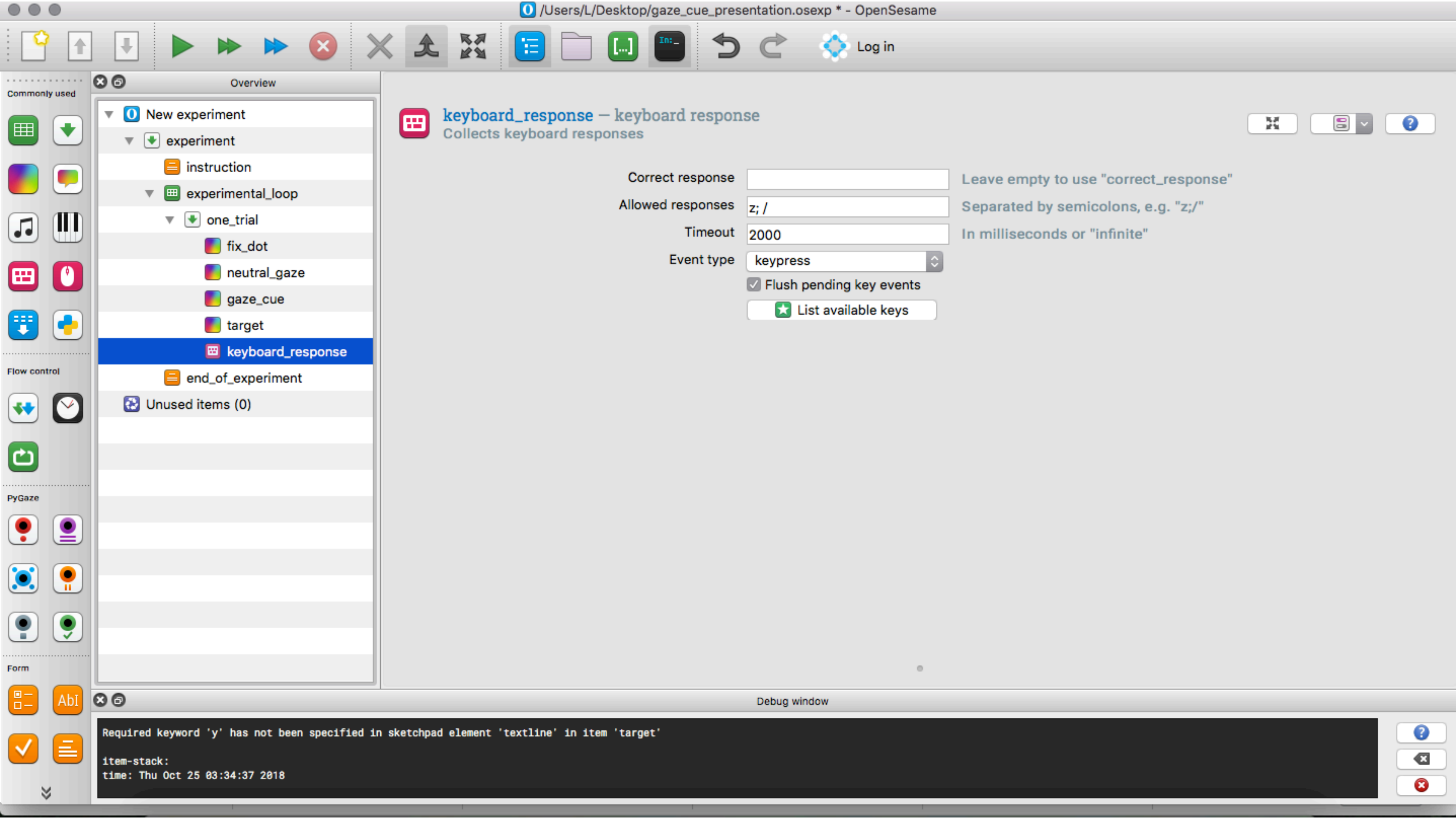
target — sketchpad
Displays stimuli

Duration keypress | Scale 1,00 x | Rotate 0° | ☑ Center | Show if always | Name auto | 1,00 x

-160,0 ☑ Grid 32 px

```
set description "Displays stimuli"
draw image center=1 file="gaze_[gaze_cue].png" scale=1 show_if=always x=0 y=0 z_index=0
draw textline center=1 color=black font_bold=no font_family=mono font_italic=no font_size=18 html=yes show_if=always text=F x="[target_pos]" y=0 z_index=0
draw textline center=1 color=black font_bold=no font_family=mono font_italic=no font_size=18 html=yes show_if=always text=X x="[dist_pos]" y=0 z_index=0
```

```
item-stack: experiment[run].experimental_loop[run].one_trial[run].keyboard_response[run]
user_triggered: True
time: Thu Oct 25 03:39:47 2018

In [1]:
```



keyboard_response — keyboard response

Collects keyboard responses

Correct response: (empty) — Leave empty to use "correct_response"

Allowed responses: z; / — Separated by semicolons, e.g. "z;/"

Timeout: 2000 — In milliseconds or "infinite"

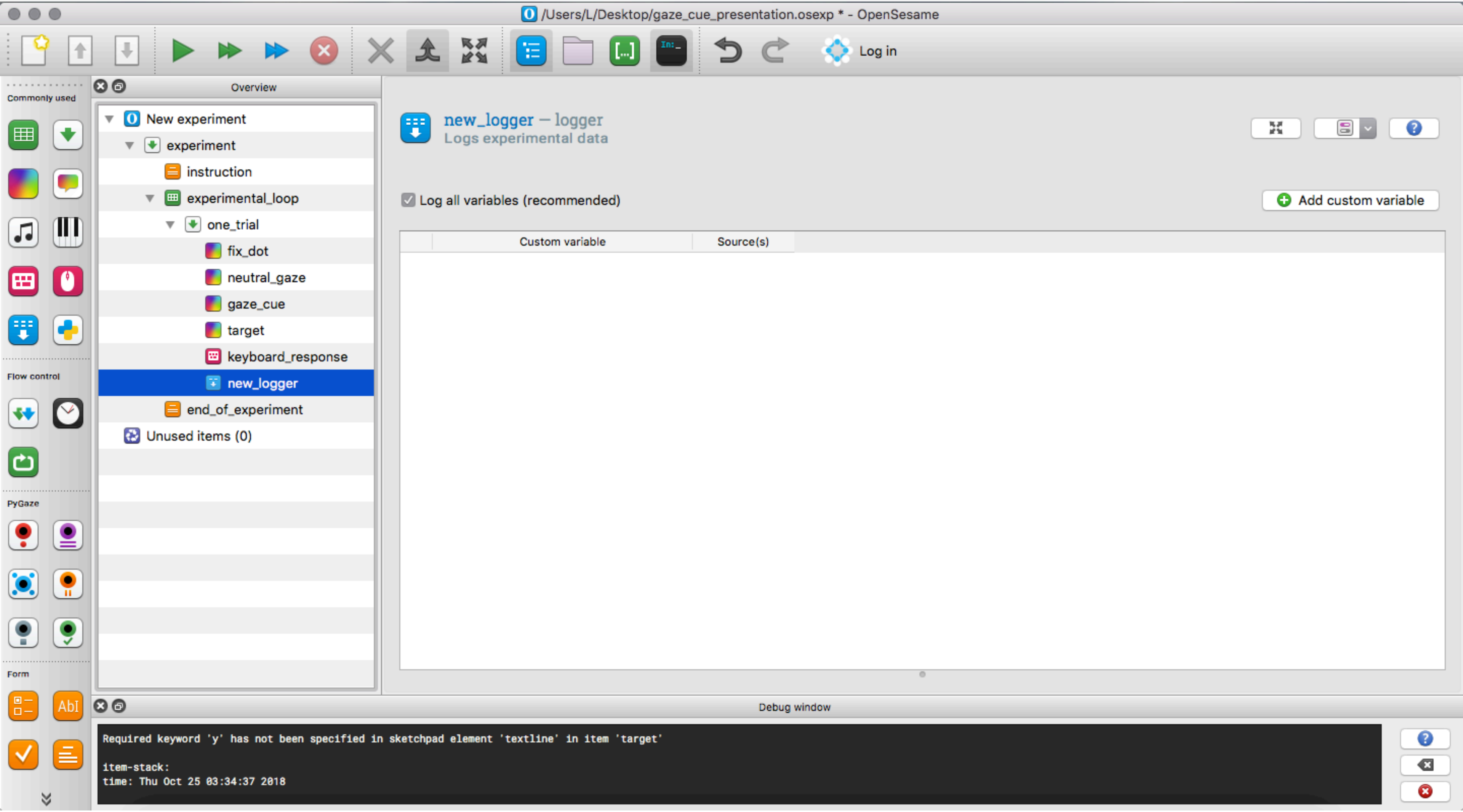
Event type: keypress

☑ Flush pending key events

List available keys

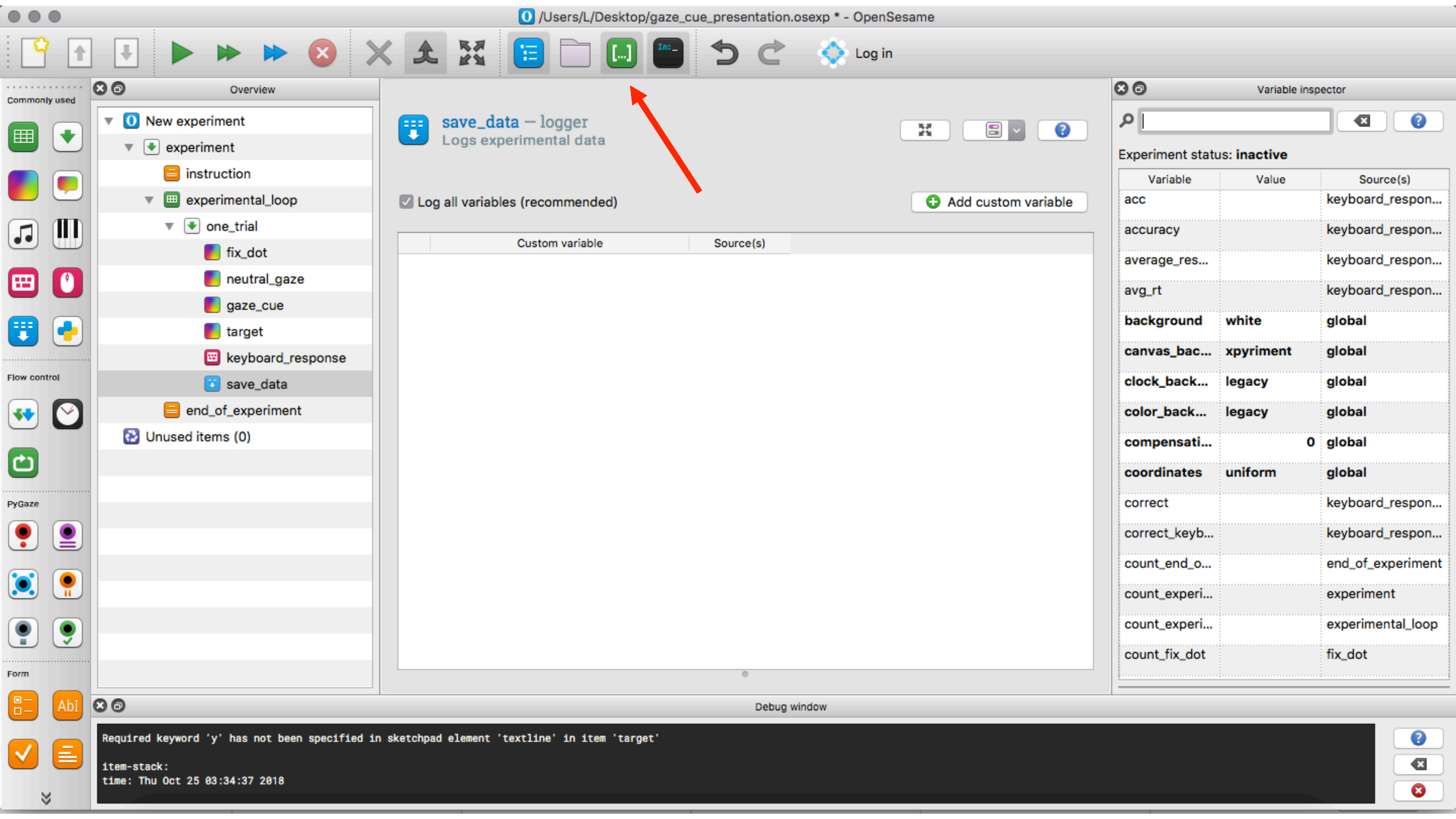
```
Required keyword 'y' has not been specified in sketchpad element 'textline' in item 'target'

item-stack:
time: Thu Oct 25 03:34:37 2018
```

Save data

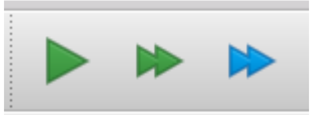


☑ Log all variables (recommended)

```
Required keyword 'y' has not been specified in sketchpad element 'textline' in item 'target'

item-stack:
time: Thu Oct 25 03:34:37 2018
```



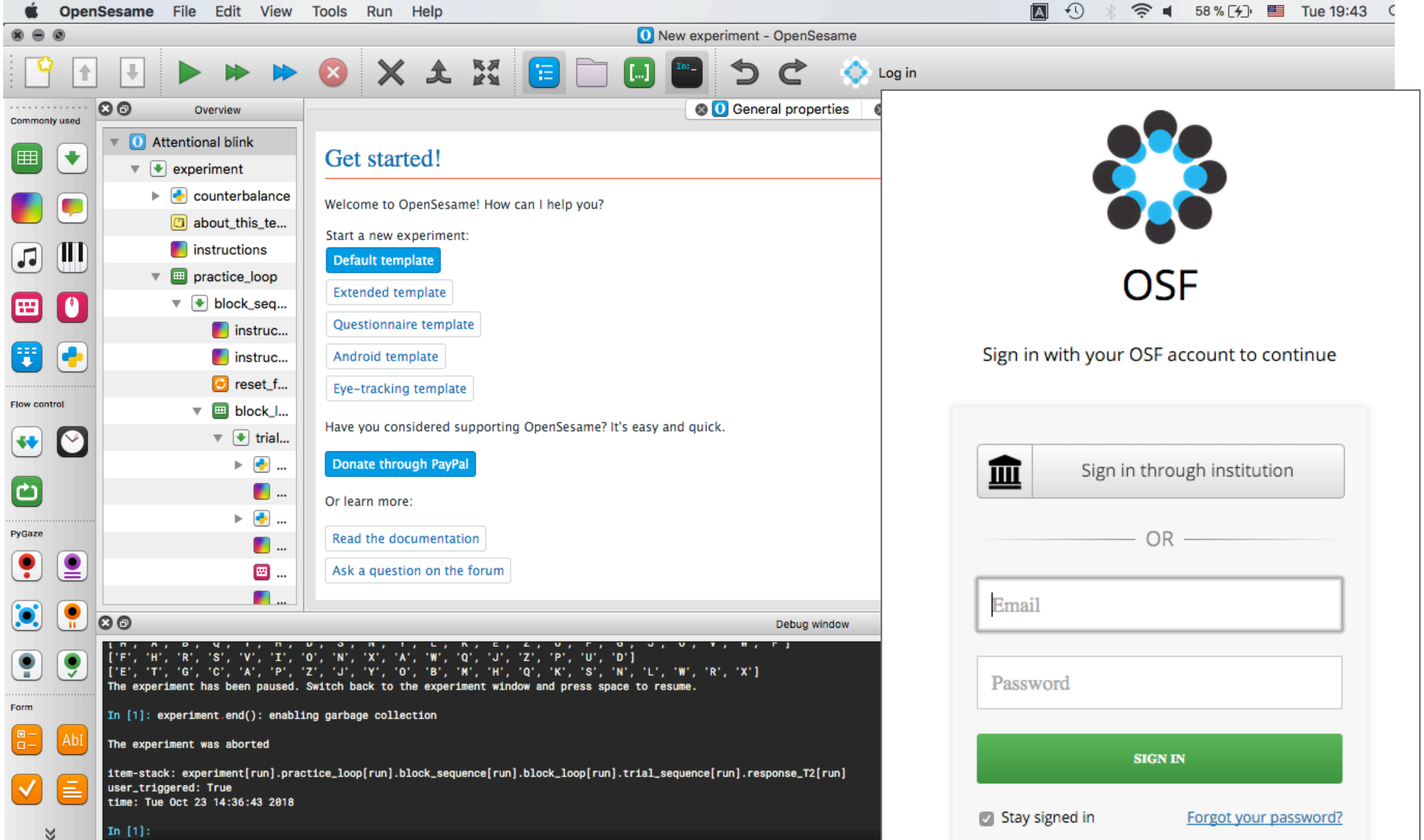
Run the experiment



Add eye tracker

1. Initialize eye tracker at the beginning of the experiment (use “advanced dummy mode” for now)
2. Add start and stop recording
3. Save log file each trail

Other options



Get started!

Welcome to OpenSesame! How can I help you?

Start a new experiment:

Default template

Extended template

Questionnaire template

Android template

Eye-tracking template

Have you considered supporting OpenSesame? It's easy and quick.

Donate through PayPal

Or learn more:

Read the documentation

Ask a question on the forum

```
['F', 'H', 'R', 'S', 'V', 'I', 'O', 'N', 'X', 'A', 'W', 'Q', 'J', 'Z', 'P', 'U', 'D']
['E', 'T', 'G', 'C', 'A', 'P', 'Z', 'J', 'Y', 'O', 'B', 'M', 'H', 'Q', 'K', 'S', 'N', 'L', 'W', 'R', 'X']
The experiment has been paused. Switch back to the experiment window and press space to resume.

In [1]: experiment.end(): enabling garbage collection

The experiment was aborted

item-stack: experiment[run].practice_loop[run].block_sequence[run].block_loop[run].trial_sequence[run].response_T2[run]
user_triggered: True
time: Tue Oct 23 14:36:43 2018

In [1]:
```

Sign in with your OSF account to continue

Sign in through institution

OR

Email

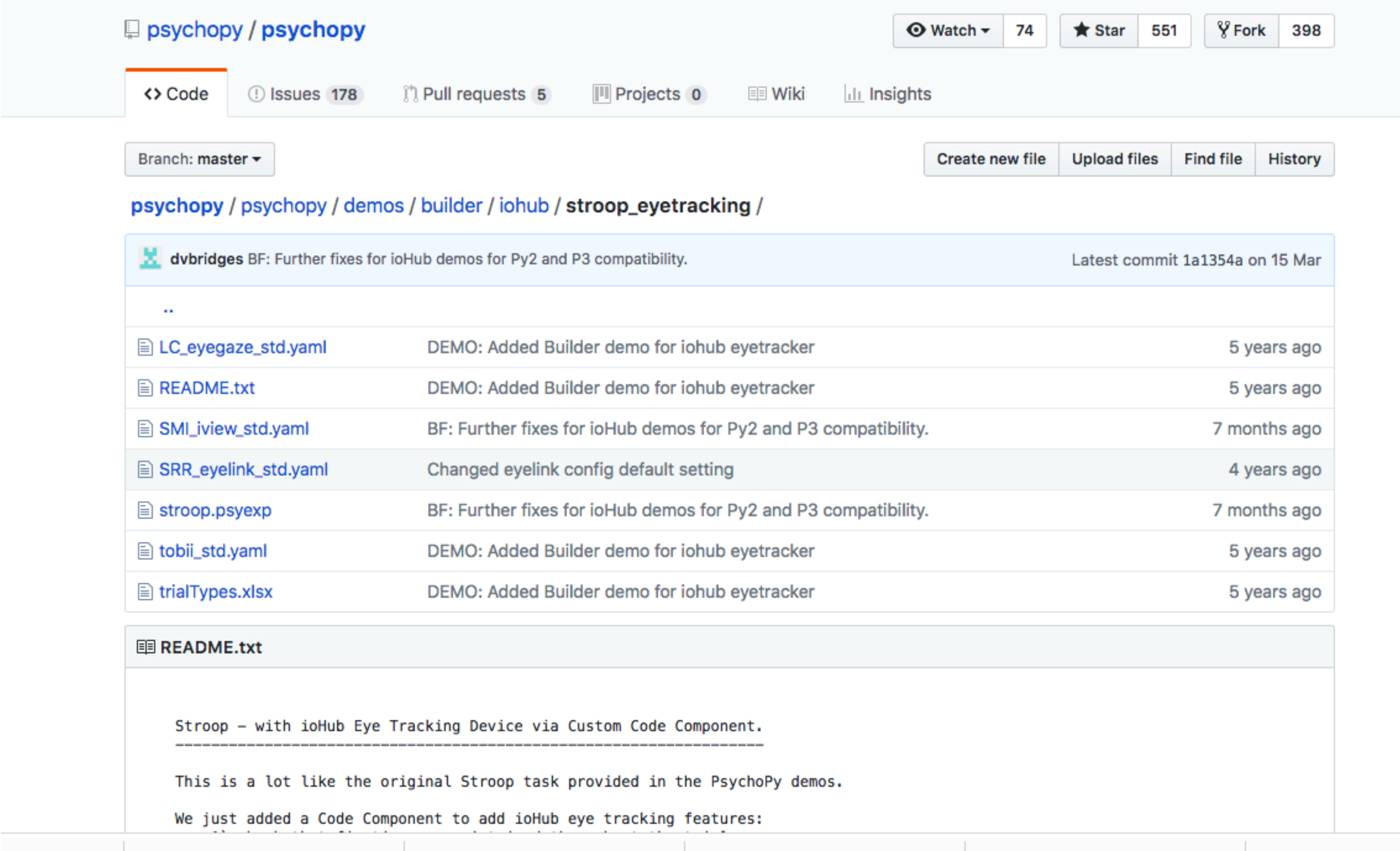
Password

SIGN IN

☑ Stay signed in

Forgot your password?

https://github.com/psychopy/psychopy/tree/master/psychopy/demos/builder/iohub/stroop_eyetracking



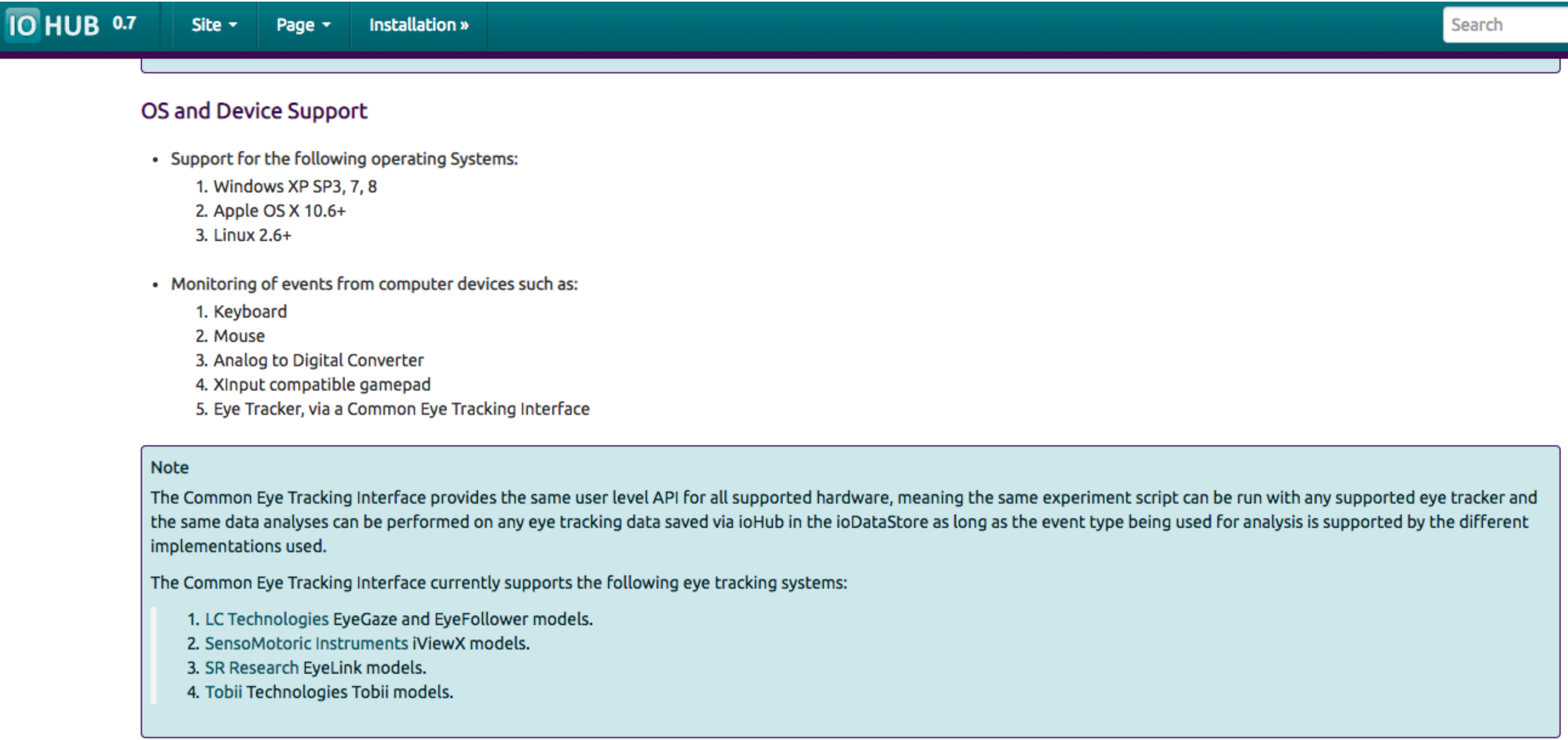
psychopy / psychopy / demos / builder / iohub / stroop_eyetracking /

dvbridges BF: Further fixes for ioHub demos for Py2 and P3 compatibility. Latest commit 1a1354a on 15 Mar

..		
LC_eyegaze_std.yaml	DEMO: Added Builder demo for iohub eyetracker	5 years ago
README.txt	DEMO: Added Builder demo for iohub eyetracker	5 years ago
SMI_iview_std.yaml	BF: Further fixes for ioHub demos for Py2 and P3 compatibility.	7 months ago
SRR_eyelink_std.yaml	Changed eyelink config default setting	4 years ago
stroop.psyexp	BF: Further fixes for ioHub demos for Py2 and P3 compatibility.	7 months ago
tobii_std.yaml	DEMO: Added Builder demo for iohub eyetracker	5 years ago
trialTypes.xlsx	DEMO: Added Builder demo for iohub eyetracker	5 years ago

README.txt

```
Stroop - with ioHub Eye Tracking Device via Custom Code Component.
------------------------------------------------------------------

This is a lot like the original Stroop task provided in the PsychoPy demos.

We just added a Code Component to add ioHub eye tracking features:
```



OS and Device Support

- Support for the following operating Systems:
  1. Windows XP SP3, 7, 8
  2. Apple OS X 10.6+
  3. Linux 2.6+

- Monitoring of events from computer devices such as:
  1. Keyboard
  2. Mouse
  3. Analog to Digital Converter
  4. XInput compatible gamepad
  5. Eye Tracker, via a Common Eye Tracking Interface

Note

The Common Eye Tracking Interface provides the same user level API for all supported hardware, meaning the same experiment script can be run with any supported eye tracker and the same data analyses can be performed on any eye tracking data saved via ioHub in the ioDataStore as long as the event type being used for analysis is supported by the different implementations used.

The Common Eye Tracking Interface currently supports the following eye tracking systems:

> 1. LC Technologies EyeGaze and EyeFollower models.
> 2. SensoMotoric Instruments iViewX models.
> 3. SR Research EyeLink models.
> 4. Tobii Technologies Tobii models.