



university of
 groningen

faculty of behavioural
 and social sciences



Using Python in OpenSesame

Session 5

Sebastiaan Mathôt



cogsci.nl/smathot



cognitivescience



[@smathot](https://github.com/smathot)



[@cogscinl](https://twitter.com/cogscinl)

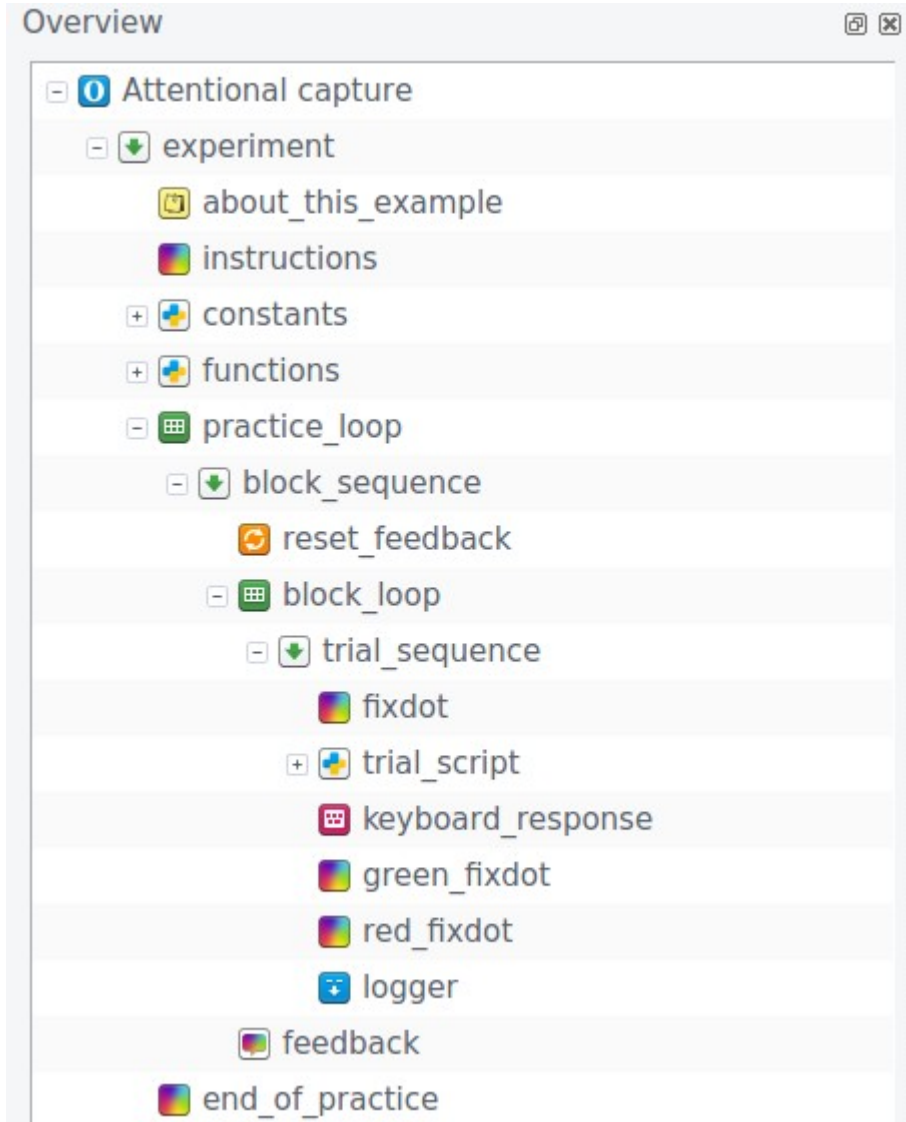


[sebastiaanmathot](https://youtube.com/sebastiaanmathot)

Items



- Yesterday, you saw that **items** are building blocks
- One item does one thing
 - Visual stimuli
 - Response collection
 - Etc.
- The **inline_script** item embeds Python script



The inline_script item



- You can include arbitrary Python code
- Scripts share a single **workspace**
 - Variables declared in one script are available in another script
 - The same goes for module imports

```
trial_script - inline script
Executes Python code

Prepare Run

1 premask_canvas.show()
2 clock.sleep(495)
3 target_canvas.show()
4
```

The inline_script item



- There are two phases
- The **prepare** phase
 - All items in a sequence are first prepared
 - Create Canvas objects, and do other time-consuming things here
- The **run** phase
 - Next all items are run
 - Keep as short as possible!

The screenshot shows a software interface titled "trial_script - inline script" with the subtitle "Executes Python code". It features two tabs: "Prepare" and "Run". The "Prepare" tab is active, displaying a list of four Python code lines in a dark-themed editor:

```
1 premask_canvas.show()  
2 clock.sleep(495)  
3 target_canvas.show()  
4
```

The inline_script item



- OpenSesame provides functions for interacting with the experiment
- These are described on the documentation site
- Let's take a look at the most common ones

Python

Python in OpenSesame

Python functions

Common functions

Variables

File pool

Items

Response history

Clock

Canvas

Keyboard

Log

Mouse

Sampler

The var object



- The var object contains experimental variables
 - Variables defined in a loop table
 - Response variables
 - Etc.
- `var.my_variable` is the Python equivalent of `[my_variable]` in the user interface



counterbalance – inline script

Executes Python code



Prepare



Run

```
1 if var.subject_parity == 'odd':  
2     var.word_response = 'z'  
3     var.nonword_response = 'm'  
4 else:  
5     var.word_response = 'm'  
6     var.nonword_response = 'z'  
7
```

The Canvas class



- The Canvas class is for presenting visual stimuli
 - Canvas elements can have names
 - And you can modify their properties later

```
new_inline_script – inline script
Executes Python code

Prepare Run

1 my_canvas = Canvas()
2 my_canvas['my_line'] = Line(-100, -100, 100, 100, color='red')
3 my_canvas.show()
4 clock.sleep(1000)
5 my_canvas['my_line'].color = 'green'
6 my_canvas.show()
7
```

The Keyboard class



- The Keyboard class is for collecting key press responses



new_inline_script – inline script

Executes Python code



Prepare



Run

```
1 # Wait for a 'z' or 'x' key with a timeout of 3000 ms
2 my_keyboard = Keyboard(keylist=['z', 'x'], timeout=3000)
3 start_time = clock.time()
4 key, end_time = my_keyboard.get_key()
5 var.response = key
6 var.response_time = end_time - start_time
```


And much more ...



- The **pool** object: accessing the file pool
- The **items** object: accessing items
- The **responses** object: response history
- The **clock** object: time functions
- The **Canvas** class: visual stimuli
- The **Keyboard** class: key presses
- The **log** object: writing to the log file
- The **Mouse** class: mouse responses
- The **Sampler** class: sound playback

Python

Python in OpenSesame

Python functions

Common functions

Variables

File pool

Items

Response history

Clock

Canvas

Keyboard

Log

Mouse

Sampler

Hands-on sessions



- First I'll show some basic Python in OpenSesame
- Then you'll work by yourself on the Extra Assignments at <https://osdoc.cogsci.nl/3.3/tutorials/wcst-python/>

