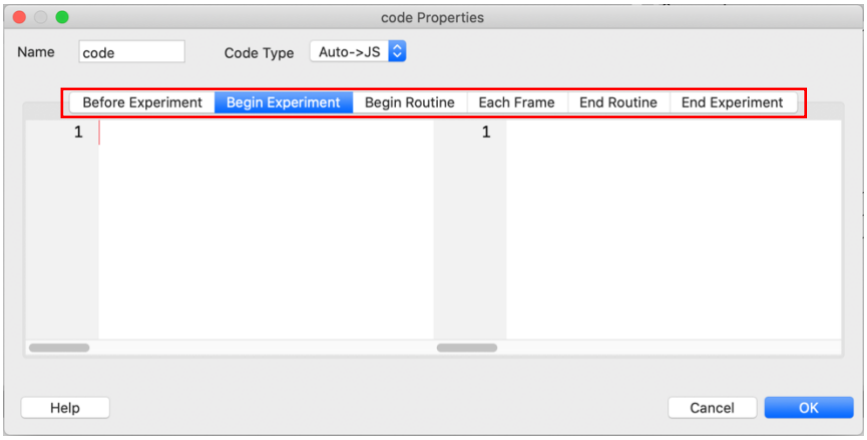
Python to PsychoJS Cheatsheet

This guide will present some useful snippets of code that can be inserted in the Custom Code component in PsychoPy. The Custom Code component window looks like this:



The left side of the window is designated for code in **Python**, which only affects experiments when they are being run in the PsychoPy app. The right side of the window is designated for code written in **PsychoJS** (javascript), which only affects an experiment when it is running in Pavlovia.

The drop-down menu next to "Code Type:" is automatically defaulted to "Auto -> JS", which means that your code in Python will automatically get translated (typically with errors!) into PsychoJS as you type complete statements. If you have a syntax error in Python, a warning will pop up on the right side when this setting is on.

Good practice would be to insert custom code on to the Python side with Auto->JS enabled and then run your experiment in PsychoPy to confirm that it works as expected. Once it works in PsychoPy, you can change the drop down to "Both," which will enable you to edit the left and right sides of the window independently.

From there, you can sync your code with Pavlovia and troubleshoot the PsychoJS side of the code as necessary. The PsychoPy troubleshooting forum (https://discourse.psychopy.org/) is your friend and will be an invaluable tool in getting your experiment to run in Pavlovia!

The tabs in the Custom Code component (outlined in red) indicate at what point in your experiment/routine that the code will be executed.

Before Experiment: A new addition to the 2020.2 update which runs code before the experiment starts (which includes before the window that your experiment is in is generated)

Begin Experiment: Code is executed right at the beginning of the experiment, which is good for defining variables

Begin Routine: Code is executed at the beginning of the routine. This is good for conditional statements where the condition in question has been determined before the current routine. For example, let's say you have a routine that presents a stimulus followed by a routine that includes a rating scale. If you would like to skip the rating routine until the stimulus has been presented X amount of times, you would insert that code in the Begin Routine tab of the rating routine. If you want to change a variable that has been used in a previous routine instead of making a new one, you can also do that in this tab.

Each Frame: Code is executed at every frame (typically 60 frames are presented each second). If you are waiting for a response in real time (keyboard press, mouse click, waiting for a certain number of stimuli to be presented, using a timer) you should use this tab.

End Routine: Code is executed after the components in the routine finish. This is good for adding a number to a counter after a component completes.

1. Continuing/ending a routine
   - You can indicate whether you want your routine to continue as planned or abort it prematurely by using the following code

Python

```python
# Continue routine
    continueRoutine = True

# End routine
    continueRoutine = False
```

PsychoJS

```javascript
// Continue routine
    continueRoutine = true;

// End routine
    continueRoutine = false;
```

2. Ending a loop
   - You can exit a loop before it completes the indicated number of repetitions by using the following code
   - Note that "nameOfLoop" is the name of your loop as it appears in the Builder flow
   - Note that this will exit the loop after the current routine finishes, so if you need to exit the routine prematurely, use the code above

Python

```python
# End a loop
nameOfLoop.finished = True
```

PsychoJS

```javascript
// End a loop
nameOfLoop.finished = true;

// For versions < 2020.2
trials.finished = true;

// or
currentLoop.finished = true;
```

3. Conditional statements
   - You may want to enter a conditional statement in your code
     ex. "If the mouse is clicked here, do this. Otherwise, do this" or "If these many stimuli are presented, continue this routine. Otherwise, don't start this routine"
   - Conditional statements in Python are very simple and there are plenty of guides online to help you navigate using if, and, or, else, and elif. These typically translate over to PsychoJS really well using the Auto -> JS option.
   - You can also include comparison functions like =, >, <, ≤, ≥, and ≠ to compare variables and responses to numbers or other variables

4. Creating a variable (to use as a counter)
   - This code creates a variable with a value of 0 that serves as a counter and adds 1 to the count under certain conditions
   - This might be useful for someone who wants to count the number of stimuli that have been presented, or count the number of correct/incorrect responses that a participant makes

Python

```python
# Create counter
count = 0

# Add 1 to the count
if # Insert condition here:
    count = count + 1
```

PsychoJS

```javascript
// Create counter
count = 0;

// Add 1 to the count
if ((// Insert condition here)) {
    count = (count + 1);
}
```

5. Generati

- This code generates a random number that can be used as a variable or as a comparison in a conditional statement

Python

```python
# Choose a random number from 1-5
randNum = randint(1,6)
```

PsychoJS

```javascript
// Choose a random number from 1-5
randNum = (Math.floor((Math.random() * ((5 - 1) + 1))) + 1);
```

6. Saving a variable that you created
   - This code saves a variable that you created within your custom code to the .csv file that is exported at the end of your experiment
     Note: Use "thisExp" and "experiment" as shown below, not the actual name of your experiment

Python

```python
# Save variable
thisExp.addData('Column name in datasheet', variableName)
```

PsychoJS

```javascript
// Save variables
psychoJS.experiment.addData('Column name in datasheet', variableName);
```

7. Aborting your experiment
   - This code will force PsychoPy to quit or end your session in Pavlovia (if for example somebody does not consent to participate in your study)

Python

```python
# End experiment
if # Insert condition here:
    win.close()
    core.quit()
```

PsychoJS

```javascript
// End experiment
if (// Insert condition here) {
    return quitPsychoJS('Type message here', false);
}
```

8. Mouse click response
   - This code lets you use a mouse click response in a conditional statement
   - Note that "mouseName" is the name of the mouse component and X and Y are the names of the clickable stimuli

Python

```python
# If mouse component clicks X component
if mouseName.isPressedIn(X):
    # Do this
# If mouse clicks Y component
elif mouseName.isPressedIn(Y):
    # Do this instead
```

PsychoJS

```javascript
// If mouse clicks X component
if (X.contains(mouseName) && mouseName.getPressed()[0] === 1) {
    // Do this
}  else {
    // If mouse clicks Y component
    if (Y.contains(mouseName) && mouseName.getPressed()[0] === 1) {
        // Do this instead
}
```

9
   - This is useful if you want to use a participant's rating on a Slider component as a condition

- You can compare the rating on the slider to a specific number, to a variable created in a custom code, or to a variable in a conditions file
- Note that “nameRating” is the name of your Slider component

Python

```python
# If rating is less than X
if nameSlider.rating < X:
    # Do this
```

PsychoJS

```javascript
// If rating is less than X
if ((nameRating.rating < X)) {
    // Do this;
}
```

Note: You can use all responses in PsychoPy in conditional statements (keyboard response, onset of a microphone response, etc.)

10. Referencing a timer

- This code can be used to reference the overall timer (globalClock) for an experiment or create a new timer at the beginning of a routine and reference it later on

Python

```python
# Initialize Timer
timer = core.Clock()

# End routine after 20s
if timer.getTime() > 20:
    continueRoutine = False

# Reference globalClock
if globalClock.getTime() == X:
    # Do this
```

PsychoJS

```javascript
// Initialize Timer
timer = new util.Clock();

// End routine after 20s
if ((timer.getTime() > 20)) {
    continueRoutine = false;
}

// Reference globalClock
if ((attn_timer.getTime() === X)) {
    // Do this;
}
```