Lab 5

Table of contents

PsychoPy basics

Lab class

In this lab, you will start to create our own experiments using PsychoPy, a piece of software developed by Jon Peirce here at the University of Nottingham. With PsychoPy, the most important thing will be that you try things out. Remember that you cannot break PsychoPy[1], so nothing can go wrong!

> < fa graduation-cap > Assessment
>
> Your knowledge of PsychoPy will be assessed by a summative PsychoPy assignment and by PsychoPy questions in the exam in January.

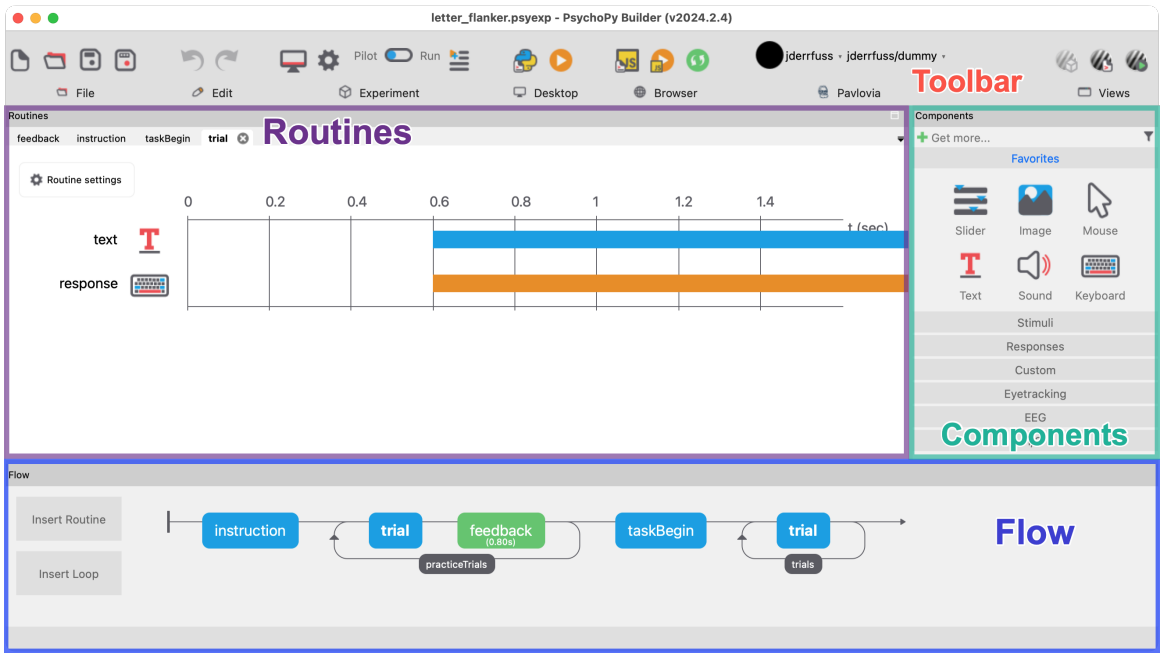
We will demonstrate how to ...

- Launch PsychoPy.
- Unzip a zip file containing an experiment and associated files.
- Open an experiment.
- Run an experiment.
- Quit an experiment.

Before we show you these, here are a few key terms you should know:

This is the so-called **Builder window**:

[1] And even if you did: You could simply download and install it again!



The Builder window has four main divisions: The toolbar, the routines panel, the components panel and the flow panel.

Here is a brief explanation of what routines, components and the flow are:

Routines

- Think of a routine as a **container or a time window** within your experiment.
- A routine itself doesn't show anything; it simply defines a time period where various elements can be presented or responses can be collected.
- For example, you would typically have a routine for your instructions and another for your experimental trials.

Components

- Components are the **actual, active elements** within a routine. These are the specific things the participant will see, hear, or interact with.
- Examples include Text, Image, and Sound stimuli, as well as response-collecting components like Keyboard, Mouse, or Microphone.
- Each component is placed on the Routine's timeline, where you define its start time and duration.

The Flow

- The flow is the **overall timeline** of your experiment. It controls the sequence of your routines.
- You arrange your routines (like "instructions," "trial," and "goodbye") in the flow panel to determine the order in which they occur.
- Crucially, the flow is where you add **loops**. A loop allows you to repeat a routine or a sequence of routines multiple times.

< fa display > Try it out

Try all of the actions yourself:

- Launch PsychoPy.
- Unzip the zip file.
- Open the experiment.
- Run the experiment.
- Quit the experiment.

Self-study

Why learn PsychoPy?

We think learning how to use PsychoPy is important for four key reasons: The first reason is that we would of course like you to be able to create experiments. However, we would argue that using PsychoPy also teaches you other fundamental skills:

- Attention to detail: When creating experiments, small things matter. E.g., a period or space in the wrong location could stop an experiment from running or make data not interpretable. Practising this attention to detail is not only relevant for creating experiments, but also for any type of data analysis you will perform during your studies.
- Staying organised: Creating experiments in PsychoPy involves keeping track of multiple files (and likely multiple versions of the same file) and making sure you are using the correct ones when running your experiment. Learning how to keep things organised on your computer is an important transferable skill that is important for other more complex tasks (such as working on a lab report or a final-year project).
- Problem solving skills: No one is born a master and when you start using PsychoPy, things might not work right away. You will likely need to hone your problem solving skills to find solutions to the problems you encounter.

We would argue that the strategies you can use to solve PsychoPy problems are ones that can be helpful in many other contexts. These strategies include:

- Breaking complex tasks down into smaller steps.
- Manipulating only one thing at a time.
- Simplifying things (e.g., by creating a simpler version of the experiment that focuses only on what is not working).
- Using feedback (or, in the case of PsychoPy, error messages) for working out what is going wrong.
- Flexibility. If solving a problem seems impossible, maybe it's worth taking a step back and trying a different approach.
- Resilience. Persisting when things don't work is really important. And it's a great feeling if they eventually do!

The remaining sections in this chapter will describe five basic PsychoPy operations:

- Launching PsychoPy.
- Opening experiments.

- Running experiments.
- Quitting experiments.
- Saving experiments.

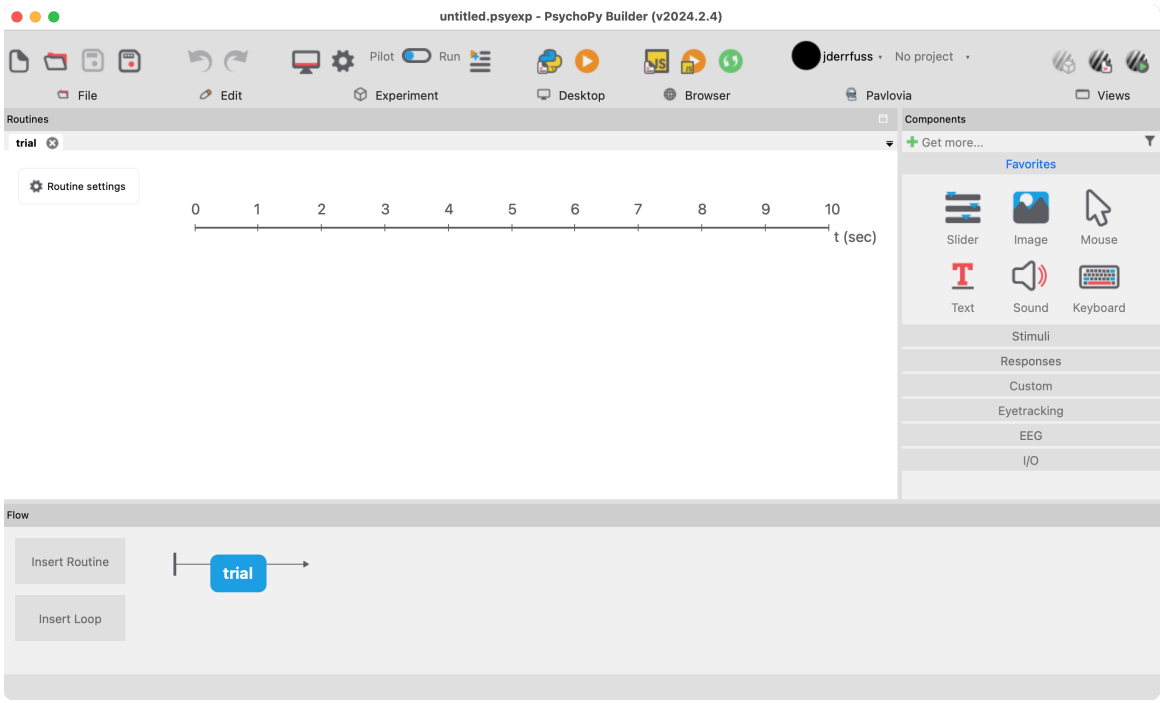
Launch PsychoPy

To launch PsychoPy, click on the PsychoPy icon:

> < fa display > Try it out
>
> Launch PsychoPy.

Depending on your operating system and hardware, PsychoPy can take a while to launch (potentially up to a minute or so).

Once PsychoPy has launched, make sure the **Builder window** is in the foreground:



Note that the next chapter will cover the Builder in more detail.

Unzip an experiment

Click here to download the ZIP file.

By default, the file will probably end up in your Downloads folder. If you use one of the iMacs in the School, we would recommend to move the file to the Desktop. If you use your own laptop, we would recommend to move the file to a suitable location on your hard drive, e.g.:

```
Year1
├── PSGY1001
│   ├── Lab5
```

When you have moved the ZIP file to a suitable location, **unzip the ZIP file** (remember that **?@sec-zip-files** explained how to do this). Once you have unzipped the file, open the folder that was created when you unzipped the file. The folder contains these files:

- `letter_flanker_input.xlsx`: This is the **input or conditions file** for PsychoPy. It defines the conditions for the experiment. Input files will be explained in more detail in a later chapter.
- `letter_flanker_instruct.png`: An **image of the task instructions**. How the instruction image is used is further explained in Section .
- `letter_flanker.psyexp`: **The `.psyexp` file is the PsychoPy experiment.** This is the file you want to open in PsychoPy.

Note: All three files must be present **in the same folder** for the experiment to run successfully.

Open an experiment

We recommend the following way to open a PsychoPy experiment file:

1. Open PsychoPy.
2. Drag and drop the `.psyexp` file onto the Builder window.

There are other options (double-click or right-click), but in our experience drag and drop works best.

⚠ macOS users

For the double-click to work under macOS, you first need to tell macOS that it should open `.psyexp` files using PsychoPy. Here is how to do this (if you follow these steps exactly, you will need to do this only once):

1. Right-click on the `.psyexp` file Get Info Open with Choose… PsychoPy.
2. You must then make sure to click on “Change all” and “OK” to confirm.

< fa display > Try it out

Open the flanker experiment.

Run an experiment

The simplest way to run an experiment is to click on the “Run” icon:

The colour of the Run icon depends on the setting for Pilot vs. Run:

If the toggle switch is set to “Pilot”, the Run icon will be orange. When set to “Run”, it is green.

For now, the key difference between these options is that Pilot mode will not run an experiment in full screen. This can make it easier to quit the experiment when something goes wrong.

> < fa display > Try it out
>
> Switch to “Run” mode (so the “Run” button turns green) and run the flanker experiment.

Quit an experiment

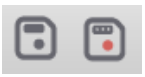
You can press the `Escape` key at any time to quit a running experiment.

If you are running macOS and pressing `Escape` has no effect, try `Cmd + Alt + Escape`.

> < fa display > Try it out
>
> Quit the flanker experiment.

Save an experiment

To save a new or modified experiment, click on the “Save” icon or the “Save as” icon in the toolbar.



Alternatively, you can press `Cmd + S` (macOS) or `Ctrl + S` (Windows).

> ! Develop and run locally, then save online
>
> When working on experiments and running them, we would recommend to always save PsychoPy experiments locally (i.e., not on OneDrive). Once you’re finished with

an experiment, make sure to save a copy on OneDrive though, so you have a back-up copy and can also access it from another computer if necessary.
If you are using one of the iMacs in the school, we would recommend to initially save your experiments in separate folders on the Desktop and to copy them to OneDrive at the end of the session. You can copy experiments to OneDrive by opening OneDrive in a browser window, navigating to the folder where you want to save the experiment, and dragging and dropping the experiment onto the browser window.

Confirmation

! Important

Please confirm you have worked through this chapter by submitting the corresponding chapter completion form on Moodle.

The PsychoPy Builder

Lab class

We will use the **PsychoPy Builder** (as opposed to the Coder) to create experiments. In this chapter, we will show you how to use the Builder to create what we like to call the **simplest possible PsychoPy experiment**. First watch how the experiment is created, then have a go yourself.

Creating the simplest possible experiment requires just four steps:

- Open PsychoPy and make sure the Builder window is in the foreground.
- Click on the Text component (the red T icon on the right).
- Click on "OK" in the window that opens.
- Click on the disk icon in the toolbar to save the experiment. (You can leave the default file name.)

That's it! Now let's run the experiment:

- Click on the Run icon (white arrowhead on orange background above "Desktop").
- Wait.
- Click on "OK" in the window that opens.

If everything worked, the sentence "Any text/including line breaks" was presented for 1 s.

Let's take this one step further: We'll present the text until a response is made. To achieve this, we need to make the following changes:

- Click on the blue timeline (or on the word "text") to open the properties window.
- Find Stop/duration (s) and delete `1.0`.
- Click on "OK".
- Click on the Keyboard component (the keyboard icon on the right).
- Delete `'y','n','left','right','space'` in "Allowed keys" (this means that PsychoPy will respond to any key you press).
- Click on "OK".
- Save the experiment.
- Run it.

That's it - the experiment should now wait for you to press a key before it quits.

> Tip
>
> **Always run PsychoPy experiments from a local copy** (but make sure to save a backup copy online - e.g., on OneDrive).

< fa display > Try it out

Create these two simple experiments.

Self-study

The multi-course meal analogy

We will use an analogy to help you understand how PsychoPy works. Think of an experiment as a multi-course meal. Then:

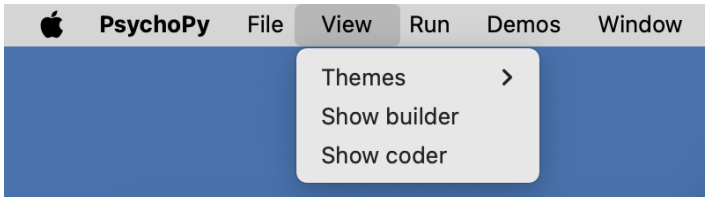
- **Routines** are individual dishes (see Section).
- **Components** are ingredients that make up individual dishes (see Section).
- The **flow** is our menu (i.e., the order in which the dishes are served).

Like all analogies, it's not perfect, but it might be a helpful starting point to thinking about experiments in PsychoPy.

Open the Builder

If you launch PsychoPy, the Builder window will be one of the windows open by default (the others being the so-called Coder and the Runner).

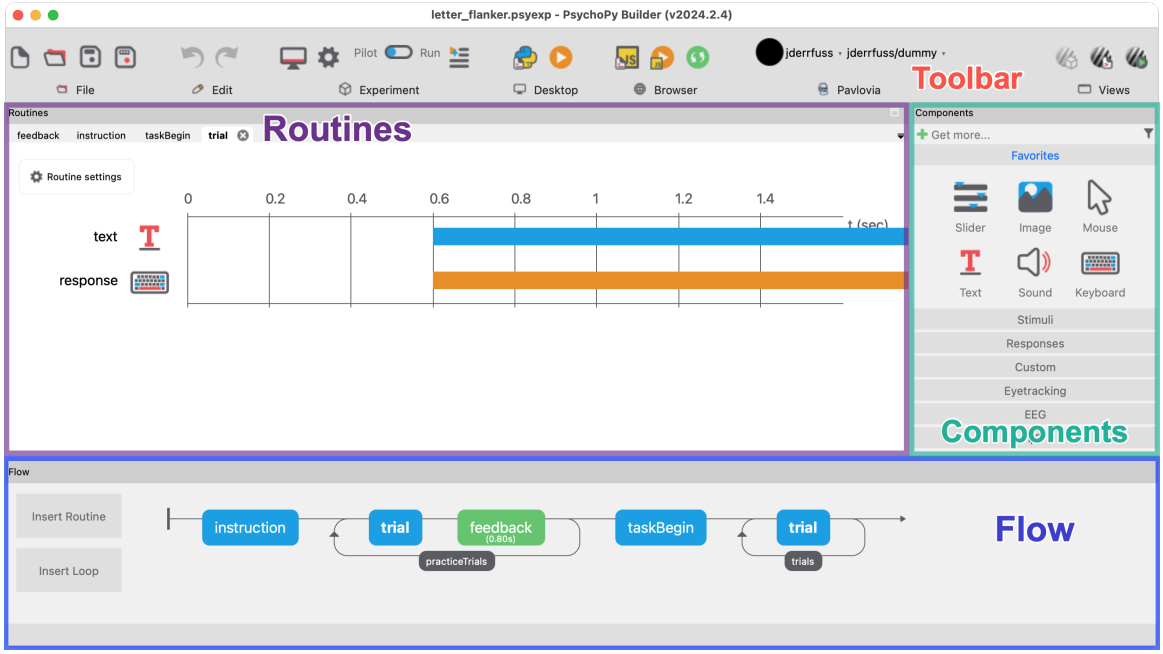
If you accidentally close the Builder window, you can re-open it by clicking on View "Show builder" in the menu bar:



Alternatively, you can click on the "Show Builder" icon in the "Views" section of the Coder or Runner toolbar, if these are still open.

The Builder window

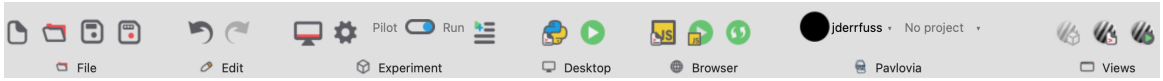
The Builder window has four main divisions: **Toolbar, Routines, Components and Flow**.



We will go over the toolbar and the flow in this chapter. Routines will be covered in Section and components in Section .

The toolbar

The toolbar is located at the top of the Builder window:



There are seven groups of icons:

- **File**: New, Open, Save, Save as.
- **Edit**: Undo and redo.
- **Experiment**: Monitor center, Experiment settings, Pilot vs. Run, Send experiment to runner.
- **Desktop**: Write Python code, Run.
- **Browser** and **Pavlovia**: Only relevant for running experiments online. We can safely ignore these for now.
- **Views**: Switch between/open Builder, Coder and Runner.

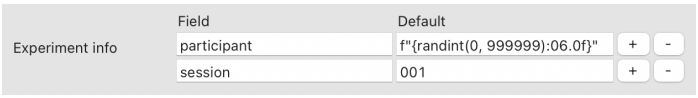
If you hover your mouse pointer over the icons, they will tell you what they do. For us, the most relevant of these icons are “Experiment Settings”, “Pilot vs. Run” and “Run”. “Pilot vs. Run” and “Run” have already been covered in Section . We will therefore focus on the “Experiment settings” below.

Experiment settings

The icon for the experiment settings is a cogwheel:

Most relevant in the experiment settings are the Basic tab and the Screen tab. The **Basic tab** is where you can change the experiment information that is part of the start-up dialogue that is presented at the beginning of an experiment.

The default setup for **Experiment info** asks about `participant` and `session`:



`f"{randint(0, 999999):06.0f}"` means that a random six-digit participant number between 000000 and 999999 will be created (this is not there for the flanker experiment, but you will come across this when you create a new experiment).

Here, you can **add fields by clicking on a plus sign** (e.g., to add fields for age and gender). You can also **remove fields by clicking on the minus sign next to a field**. Note that you should leave the `participant` field untouched. PsychoPy expects to find this as is (i.e., including the lower case first letter)!

The **Screen tab** gives you access to settings such as the **background colour**. The default background colour is grey:

The three numbers correspond to, in this order, the intensity of red, green and blue (RGB) channels ranging from -1 to 1. Thus, other background colours can be defined in this way:

- black: `$[-1, -1, -1]`
- white: `$[1, 1, 1]`
- red: `$[1, -1, -1]`
- green: `$[-1, 1, -1]`

The easiest way to change the background colour though is to click on the colour picker:

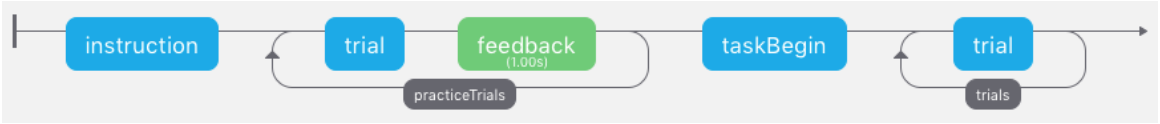
You can then choose a named colour (such as “black” or “cornflowerblue”) simply by clicking on the colour name.

The flow

The flow depicts the order in which routines will be presented as coloured rectangles ordered from left to right. **The colour of a routine indicates whether or not it has a fixed length**. If it has a fixed length, it is displayed in green. If not (e.g., if the trial only ends when a response key is pressed), it is displayed in blue.

Some routines might have loops around them (dark grey rectangles with curved arrows). This usually[1] means that the routine(s) in the loop will be repeated multiple times (we will talk about this in more detail in Lab 6).

In our flanker task, we start with the routine `instruction`. This routine will be followed by the routine `trial`, which will be followed by the routine `feedback`, etc.:



When routines are repeated in a loop, typically something changes from one repetition to the next. If we were to stretch our analogy a bit, it could be the waiter coming over to your table repeatedly and on each occasion serving a new type of bread: "Here is a slice of bread with walnuts." "Here is a slice of bread with pumpkin seeds." "Here is a slice of bread with sunflower seeds." "Here is a slice of bread with flax seeds." You get the idea. In our flanker task, the loops update the stimulus from one trial to the next. For example, the first stimulus might be `HHHHH`, followed by `SSHSS`, `HHSHH`, `SSSSS` and so on. We are going to have a closer look at loops next week.

Also, note that dishes can be served again. That is, **routines can be re-used**. In our task, we use the routine `trial` twice. What would be a no-go in a restaurant is a great feature in PsychoPy. If practice and experimental trials require the same components with the same properties, you need to create your trial routine only once and you can then simply reuse it when you need it again!

Confirmation

> **!** Important
>
> Please confirm you have worked through this chapter by submitting the corresponding chapter completion form on Moodle.

[1] A loop has a specific number of repetitions associated with it. If the number of repetitions is set to `0`, this can be used to stop a routine or multiple routines from running. This is useful for debugging experiments.

Routines

Self-study

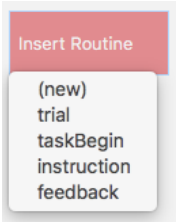
In our multi-course meal analogy, routines correspond to individual dishes. What a dish tastes like depends on the ingredients. Analogously, **what a routine does depends entirely on its components** (see Section). In a way, a routine is just a container used to combine components.

Just as some dishes will have very few ingredients, some routines might consist of just one or two components. Others will have many more components, for example in cases where you want to present multiple independent stimuli.

Adding and removing routines

You can **add a new routine** to an experiment by clicking on **"Insert Routine" and "(new)" in the flow**. Newly created routines will be empty and are waiting for you to add components to them.

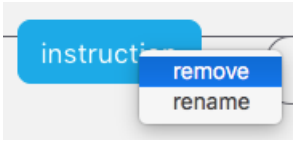
Note that "Insert Routine" can also be used to add already existing routines to the flow:



After clicking on "Insert Routine", you can add a routine to a specific position in the flow by clicking on the circle that appears when you move the mouse pointer over the flow:

You can click on "CANCEL Insert" if you're not yet sure where in the flow you would like to place it.

To **remove a routine from the flow**, right-click on the routine and select remove:

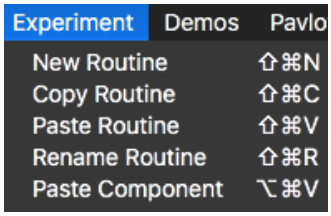


Note that this only removes the routine from the specific location in the flow, but not from the experiment or from another location in the flow.

To **remove a routine from an experiment completely**, click on the < fa circle-xmark > next to the routine's name. This will automatically remove the routine from the flow as well.

Copying and pasting routines

Note that you can also **copy and paste existing routines** (including from one experiment to another):



This can be a great time-saver!

Routines in our flanker task

Our flanker experiment has four routines: `feedback`, `instruction`, `taskBegin` and `trial`. Note that each routine has a separate tab. The routine `trial` is currently selected as indicated by the lighter shading and the < fa circle-xmark >:

In this flanker task, flankers and target appear at the same time. Therefore, only one Text component is required. This Text component might then present the stimulus `HHSHH`. What researchers sometimes do is give the flankers a head start. In this case, the flankers appear earlier than the target. Now, your routine needs *two* Text components. One to present the flankers, e.g. `HH  HH`, and the other to present the target, e.g. `S`.

Naming routines

When you create a routine, you need to give it a name. There are some **rules you must follow when naming routines**:

- Use only letters, numbers, and underscores to name them (e.g., you must not use spaces!).
- Always begin the name with a letter (numbers and underscores can only come later).
- You cannot reuse names (once you have labelled a routine or component `stim`, you cannot give another routine or component the same name).

PsychoPy will usually warn you if you try to use a name that is not possible.

Note that it is good practice to **give routines names that indicate what they do**. For example, our routine presenting the instructions is called `instruction`. If it was called `trial_2`, it would be rather unclear what its role is.

Confirmation

> **!** Important
>
> Please confirm you have worked through this chapter by submitting the corresponding chapter completion form on Moodle.

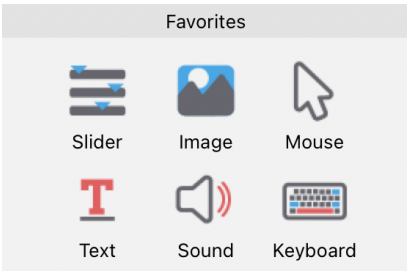
Components

Self-study

Components are the ingredients that make up routines. Some of the main functions of components are **presenting stimuli and recording responses**.

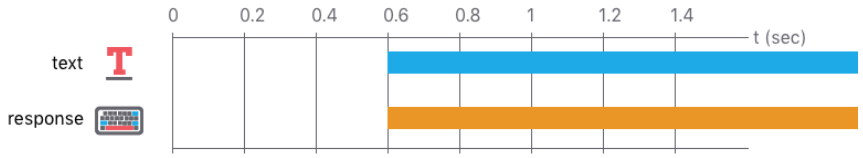
Types of components

Under “Favorites”, you can find the most frequently used components:



Additional components can be found by clicking on the appropriate component groups:

The `trial` routine in our flanker task has two components: a **Text component** and a **Keyboard component**:



You can **identify the type of component based on the icon**: The Text component’s icon is a **red T**. The Keyboard component’s icon a **keyboard**.

The **Text component presents text** on the screen and the **Keyboard component records responses** from the participant (in particular, their RT and, if set up accordingly, their accuracy).

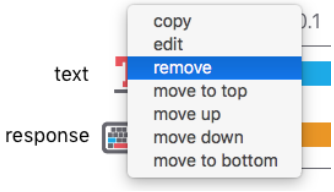
The **names of components** are chosen by the person setting up the experiment (note the naming constraints described below though). In the example above, the component names are `text` and `response`.

Components have a number of properties. For example, these define the timing, position, size, and orientation of a stimulus. We will have a closer look at some of these properties in Section .

Adding and removing components

To **add a component** to a routine, simply click on the component (drag and drop does not work here!).

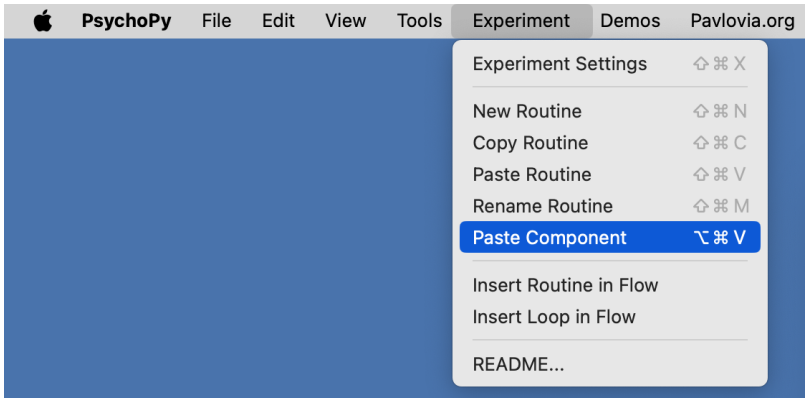
To **remove a component** from a routine, right-click on the component and select "remove":



Copying and pasting components

To **copy a component**, right-click on the component and select "Copy".

To **paste a component**, click on "Experiment" in the menu bar and then on "Paste Component":

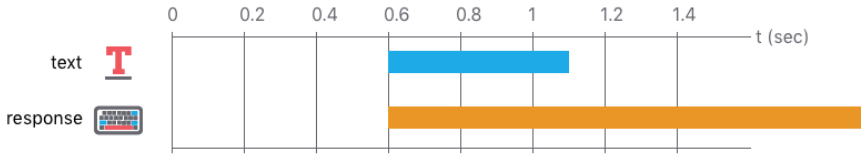


Component timing

The **blue and orange bars** next to the components represent information about timing:

- Both components start 0.6 s into the trial. That means that the stimulus presentation will begin 0.6 s into the trial and, at the same time, the Keyboard component will start measuring respone time. The first 0.6 s of the trial will simply consist of an empty screen.
  - That both components start at the same time is important to ensure that the time measured by the Keyboard component will correspond to the response time (i.e., the time since stimulus onset).
  - It is important that the onset of the Keyboard component coincides with the onset of the imperative stimulus (i.e., the stimulus to which participants should respond).[1]
  - That fact that the two components only start 0.6 s into the trial also introduces a short gap between trials[2] and thus avoids that a new stimulus is presented immediately after a response.
- Both components extend beyond the "t (sec)" label. This means that their duration is infinite (however, as experiments lasting infinitely long tend to be unpopular with participants, you can tell PsychoPy to end a routine when a response is given!).

If we change the duration of the Text component to 0.5 s, the timeline will change accordingly and the blue bar now has a length corresponding to 0.5 s:



The **colour of the component timeline** refers to the type of component used:

- **Blue** is typically used for components that present stimuli or collect data continuously (e.g., camera or microphone).
- **Orange** is used for components that expect active, discrete responses from participants.

Naming components

The same rules as for naming routines apply.

Note that it is good practice to **give components names that indicate what they do**.

[1] If that is not the case, you would need to manually correct RTs later.
[2] To be precise, a so-called response-stimulus interval (RSI).

Confirmation

> **!** Important
>
> Please confirm you have worked through this chapter by submitting the corresponding chapter completion form on Moodle.

Component properties

Self-study

This chapter covers **three key PsychoPy components and their properties: the Text, the Image and the Keyboard component**.

Where do properties fit into our multi-course meal analogy? Here's one way to think about it: **Components** are ingredients that make up individual dishes. **Properties** describe how to prepare the ingredients (e.g., what size to chop them and how long to boil them). In a PsychoPy experiment, properties determine things such as stimulus sizes, colours and positions.

The Text component

Figures 1 to 4 show the basic, layout, appearance and formatting properties, respectively, of our Text component in the routine `trial`.[1]

Show/hide the figures

[1] Usually, you will not need the other two tabs that are not shown here. Feel free to have a look at them though.

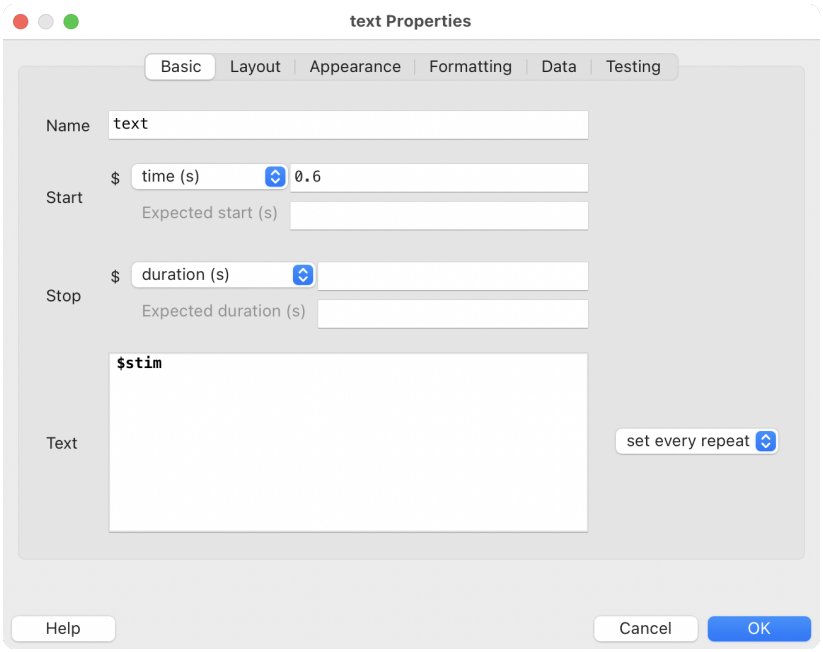


Figure 1: Basic properties of a Text component.

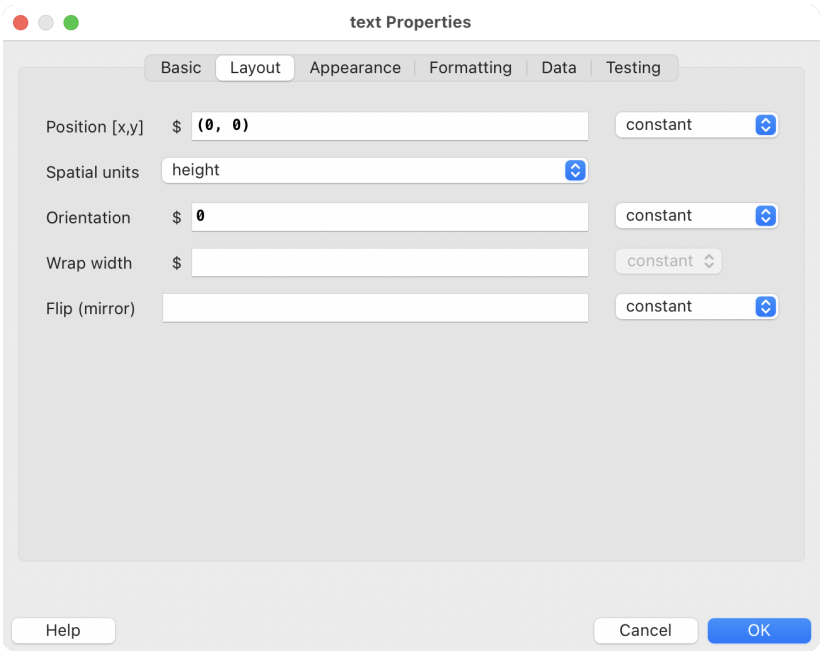


Figure 2: Layout properties of a Text component.

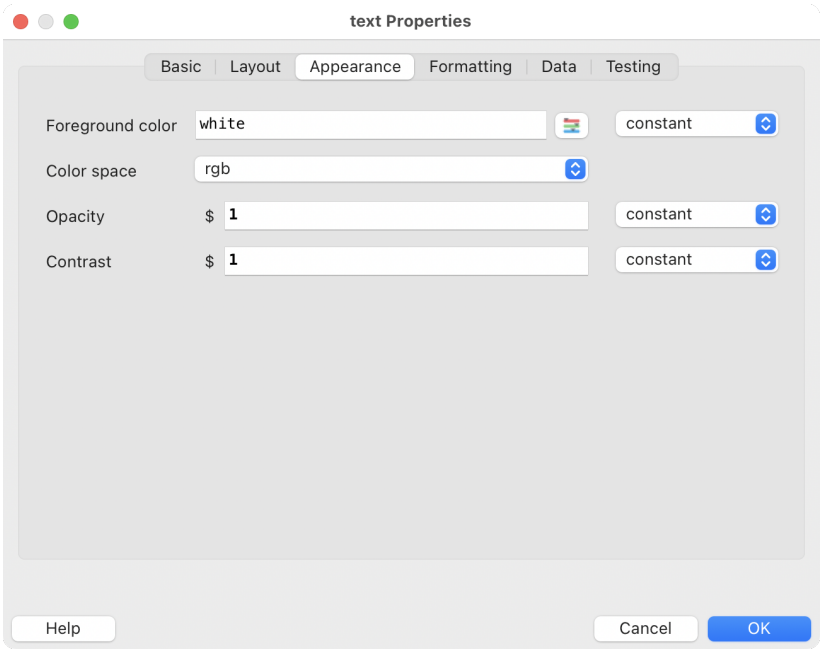


Figure 3: Appearance properties of a Text component.

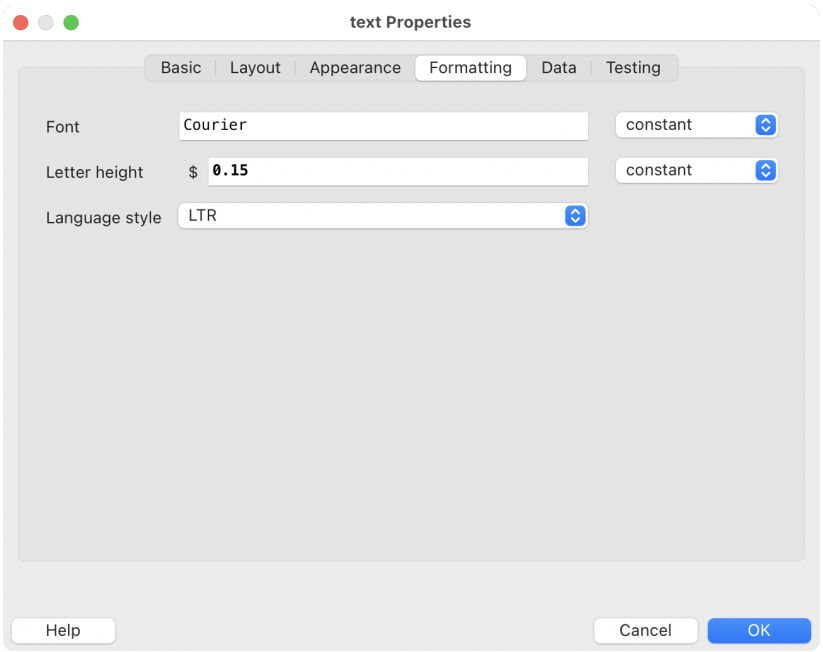


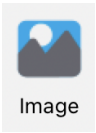
Figure 4: Formatting properties of a Text component.

Most of the properties are probably self-explanatory. A few comments though:

- Letter height in our example is 0.15. Without looking at the units used, you don't know what 0.15 refers to. Information about the unit can be found on the Layout tab. In our

example, it says `Spatial Units: height`. This means that the size of the letters will be approximately[2] 15% of the screen height.
- The position is `(0, 0)`. In PsychoPy, these are the x and y coordinates for the centre of the screen. You can learn more about this topic on this PsychoPy help page about units and using this interactive app illustrating PsychoPy units.
- We'll cover the exact meaning of `$stim` next week. Very briefly, together with the setting "set every repeat", it means that the stimulus will be updated from one trial to the next.[3]

The Image component



Figures 5 and 6 show the most relevant properties of our Image component in the routine `instruction`.[4]

Show/hide the figures

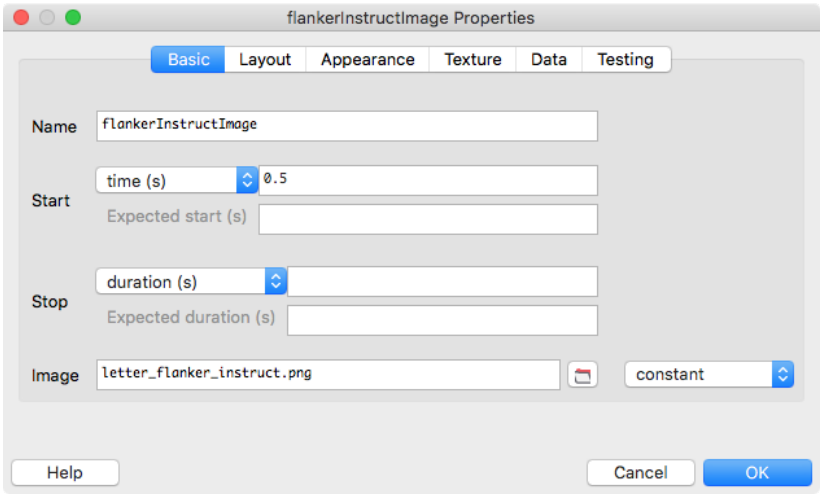


Figure 5: Basic properties of an Image component.

[2] Height will vary for individual letters. E.g., an "a" won't be as high as a "d".
[3] For comparison purposes, have a look at the Text component from the routine `taskBegin`. This has text that is set to "constant".
[4] Again, we won't need the other tabs.

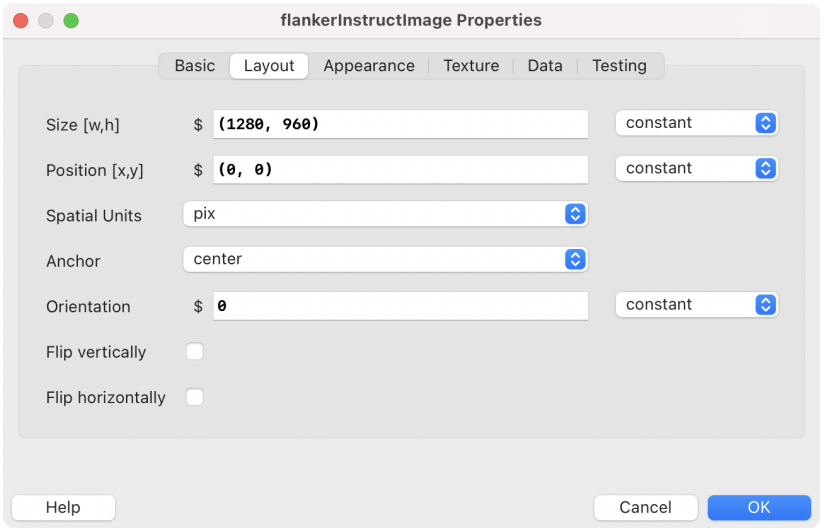


Figure 6: Layout properties of an Image component.

Again, a few comments:

- "Image" refers to a file name on your computer. In our example, this file is called `letter_flanker_instruct.png`. The image file must be located in the same directory as the experiment itself (otherwise, it would be necessary to specify the path to the image).
- The size of the image on the screen will be 1280 x 960 pixels (see "Size" and "Spatial Units").
- Using the "Orientation" property, images can be rotated.

Note that leaving the size field empty will display the image in its original size in pixels.[5]

The Keyboard component

Figures 7 and 8 show the most relevant properties of our Keyboard component in the routine `trial`.

Show/hide the figures

[5] This does not work when running an experiment online with Pavlovia though.

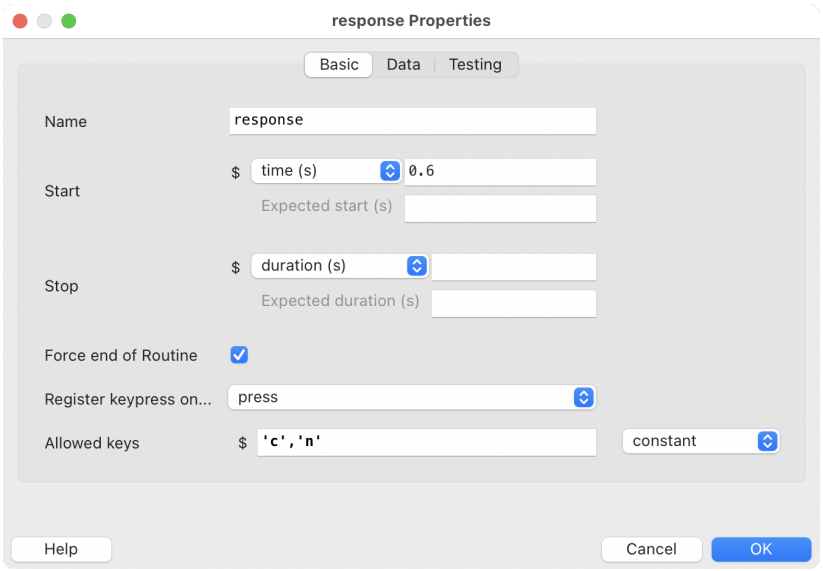


Figure 7: Basic properties of a Keyboard component.

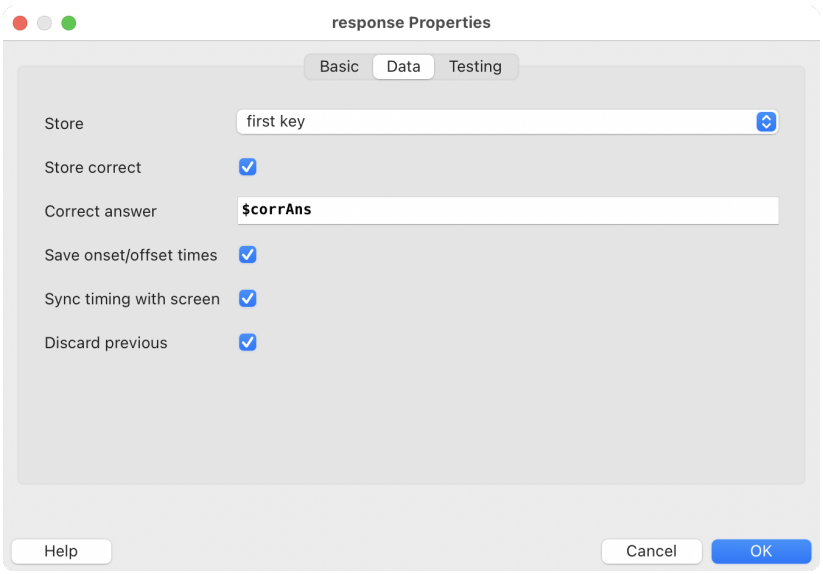


Figure 8: Data properties of a Keyboard component.

The properties of the Keyboard component deserve some more detailed comments:

Force end of Routine: If this is ticked and the participant presses one of the allowed keys, the trial will end.

Allowed keys: These are the keys that will be registered by PsychoPy. If a non-allowed key is pressed, PsychoPy will not know about this event. To allow all keys on the keyboard, leave the "Allowed keys" field empty.

Store: The default choice for storing responses is "last key". However, "first key" might be a

better choice (if a response is incorrect, you will store the incorrect key and thus know that the response was incorrect). Note, however, that if you choose “Force end of Routine”, the result for both settings will be the same as the first response will end the trial and the first key will therefore also be the last key.

Store correct: If ticked, PsychoPy will store whether or not the answer on a given trial was correct.

Correct answer: Of course, PsychoPy cannot know what the correct answer on a given trial is. Therefore, you must supply PsychoPy with this information. `$corrAns` makes PsychoPy look for a column with that name in an input file you provide PsychoPy with (more about this next week).

For the remaining properties, simply leave the default values.

Confirmation

> **!** Important
>
> Please confirm you have worked through this chapter by submitting the corresponding chapter completion form on Moodle.

Formative PsychoPy quiz

Self-study

This is a formative PsychoPy quiz about content covered in Lab 5. You might find it helpful to return to these questions when revising for the January exam.

This section contains interactive content which is not available in the PDF version. Please visit the online version to see it.

Stroop task example

Self-study

Here is a video of me building a Stroop task from scratch. We will cover some of the things I am showing only next week, but I thought it might be useful to have access to the video now if you would like to see in more detail how to interact with PsychoPy. The PsychoPy version I am using is an older one, but all the key things I am showing are similarly implemented in the current version.

This section contains content which is not available in the PDF version. Please visit the online version to see it.

Explore, apply, reflect

Lab class/ Self-study

Please complete as many exercises as you can during the lab class. If you don't finish them all, complete the remaining ones as part of your self-study.

Please save each exercise in a separate folder. This will make it easier to track your work and restore a previous version if something goes wrong with a later attempt.

Exercise 1

Start by creating a new experiment. To do so, click on the "Create a new experiment file" icon in the toolbar. This will open a new Builder window with an empty routine called `trial`.

Make PsychoPy display the text "Hello world!" for 3 seconds. Once this works, change the background colour of the screen to black.

> How do I present "Hello world!"?
>
> Use a Text component.

> How do I make the trial stop after 3 s?
>
> Pay attention to the "Stop" property of the Text component.

> How do I change the background colour?
>
> Check "Experiment Settings" → "Screen".

You can watch a video of the solution below (to watch the video, make sure you are logged into Office 365). The PsychoPy version I am using is an older one, but all the key things I am showing are similarly implemented in the current version.

This section contains content which is not available in the PDF version. Please visit the online version to see it.

Exercise 2

Please don't forget to create a new folder for this exercise. You can copy your experiment from Exercise 1 to this folder, and modify it.

Make PsychoPy display the text “Hello world!” 1 second after the start of the experiment. Present the text for an infinite amount of time. End the presentation only when the key “y” or the key “n” is pressed on the keyboard (but not another key).

> The timing isn’t right.
>
> Pay attention to the “Start” and the “Stop” properties of the Text component.

> The trial stops for other keys than “y” or “n”.
>
> Check “Allowed keys” in your Keyboard component.

> The trial does not stop.
>
> The trial will only end if “Force end of routine” is ticked and a key that is an “Allowed key” is pressed.

You can watch a video of the solution here:

This section contains content which is not available in the PDF version. Please visit the online version to see it.

Exercise 3

Download the supporting material for this exercise. This is a zip file containing an image of a cat and a dog. Unzip the file and place the images in your `exercise3` folder.

Click here to download the ZIP file.

Present the cat in the left half of the screen for 5 seconds. Two seconds after the presentation of the cat started, present the dog for three seconds in the right half of the screen. Both animals should be on screen together for 3 seconds. Present the pictures using their original size. (Make sure to open them in Preview so you know what they should look like.)

Once this works, rotate the images so they are presented upside down.

> The experiment does not run.
>
> Are you sure the images are in the same folder as your PsychoPy experiment? Are the file names in your image components identical to the file names in your folder (including capitalisation)?

> How do I change the location of the pictures on the screen?
>
> You will need to change the x variable of the Position property. If you use `norm` units, a sensible setting for the cat would be `[-0.5, 0]`. If you use `pixel` units, appropriate settings depend on your screen resolution. A sensible setting for a full HD screen (i.e., 1920 x 1080 pixels) would be `[-480, 0]`. In fact, this would be exactly the same location as `[-0.5, 0]` in `norm` units.

The pictures are squashed or stretched.

Simply leave the Size field of the Image component empty. This will present the image in its original size.

The rotation does not work.

Note that the Orientation field of an image component requires the number of degrees you would like to rotate the image by.

You can watch a video of the solution here:

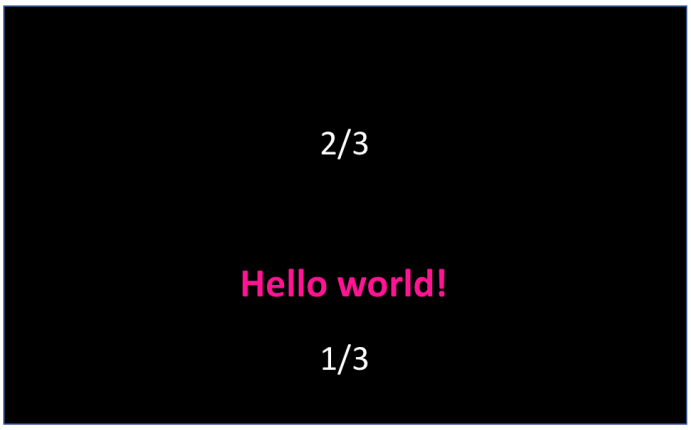
This section contains content which is not available in the PDF version. Please visit the online version to see it.

Challenge exercises

Exercises 1 and 2

Change the text colour to deep pink. Set the colour using the colour name.

Then present the text at 1/3 of the height of your screen:



Try height units as well as norm units for setting the text location. Note that the idea is to **calculate** the correct position, not to eyeball it!

How do I do this using height units?

Using height units, the total height of the screen is 1. So, the text must be placed 0.33 height units from the bottom of the screen. Using height units, the bottom of the screen is -0.5. So, the text must be placed at $-0.5 + 0.33 = -0.17$.

How do I do this using norm units?

Using norm units, the total height of the screen is 2. So, the text must be placed 0.66 norm units from the bottom of the screen. Using height units, the bottom of the screen is -1. So, the text must be placed at $-1 + 0.66 = -0.33$.

Exercise 3

Present the cat in the upper left-hand corner and the dog in the lower right-hand corner of the screen:

Note that the red outlines are only there to indicate where the images should be located. These do not need to be present when running the experiment.

How do I place the images in the corners?

The easiest way to achieve this is to use the “Anchor” image layout property.

References