Introduction to Expyriment

Programming Psychology Experiments (CORE-1)

Barbu Revencu & Maxime Cauté

Session 2 | 17 September 2025

The plan for today

1. Finish last week’s exercises (20′)
2. Your feedback (5′)
3. Introduce expyriment (15’)
4. Start coding with expyriment (50′)

Last week’s exercises

Tasks for you

```
Barbu@Mac % cd your-path/Programming/Assignments/Week-1/Exercises
```

Those of you who solved Exercise 1.1 only: Solve the next exercises

```
Barbu@Mac Exercises % python Exercise-1.1.py
```

Those of you who only solved Exercise 1: Solve Ex. 2–7 in VS Code

Those of you who solved Exercises 1–7: Raise your hand, we will come and look at your solutions

When done:

```
Barbu@Mac Exercises % cd ..\..
Barbu@Mac Assignments % git add .
Barbu@Mac Assignments % git commit -m "Week 1 Exercises"
Barbu@Mac Assignments % git push origin
```

Difficulty of Week 1’s assignments

Fill in the form at https://forms.gle/TPDjfrC3Ejww1q26A

Admin stuff

Assignments

Each week, you are expected to submit your assignment solutions **twice**

1. At the **end of each session** (5 minutes before class ends)
2. By **Sunday at 12:00 pm** for the exercises not completed in class

Both submissions count toward your evaluation

Solutions submitted **after the deadline** will **not** be considered

Our own solutions will be posted on GitHub every Monday

Discord channel

Join at **https://discord.gg/7HYSf9UU**

Use it to **ask questions** about assignments when you get stuck

Don’t hesitate to **answer other people’s questions yourself**

We will also use it to **provide feedback** on your assignments

Use your full name and (only if you’re comfortable) upload a photo

Expyriment

What is expyriment?

A Python library for designing and running psychology, neuroscience, and psychophysics experiments

It’s meant for researchers who need to **present stimuli** (text, images, sounds) and collect responses (e.g., key presses) **with good timing precision**

Pros of expyriment

A **clean and simple** psychology experiment generator, which promotes good programming practices (readability)

It relies on Python, so it aims to be **reproducible** across platforms (we'll see about that!)

It allows researchers to **focus on the high-level, abstract structure** of experiments without having to code low-level timing or graphics routines themselves

Cons of expyriment

It relies on Python, so it's **not possible to run remote online experiments** (for this, you will learn jsPsych later on in the course)

It has a **small user community**, which means that there are not many demonstrations/examples on the web (the interface, however, is very well documented)

> *Note*: This also means that **LLMs will often hallucinate** when prompted about expyriment since the training data is sparse

What does this code snippet do?

```
fixation = stimuli.FixCross()
circle = stimuli.Circle(radius=50)

fixation.present()
clock.wait(1000)
circle.present()

keyboard.wait()
```

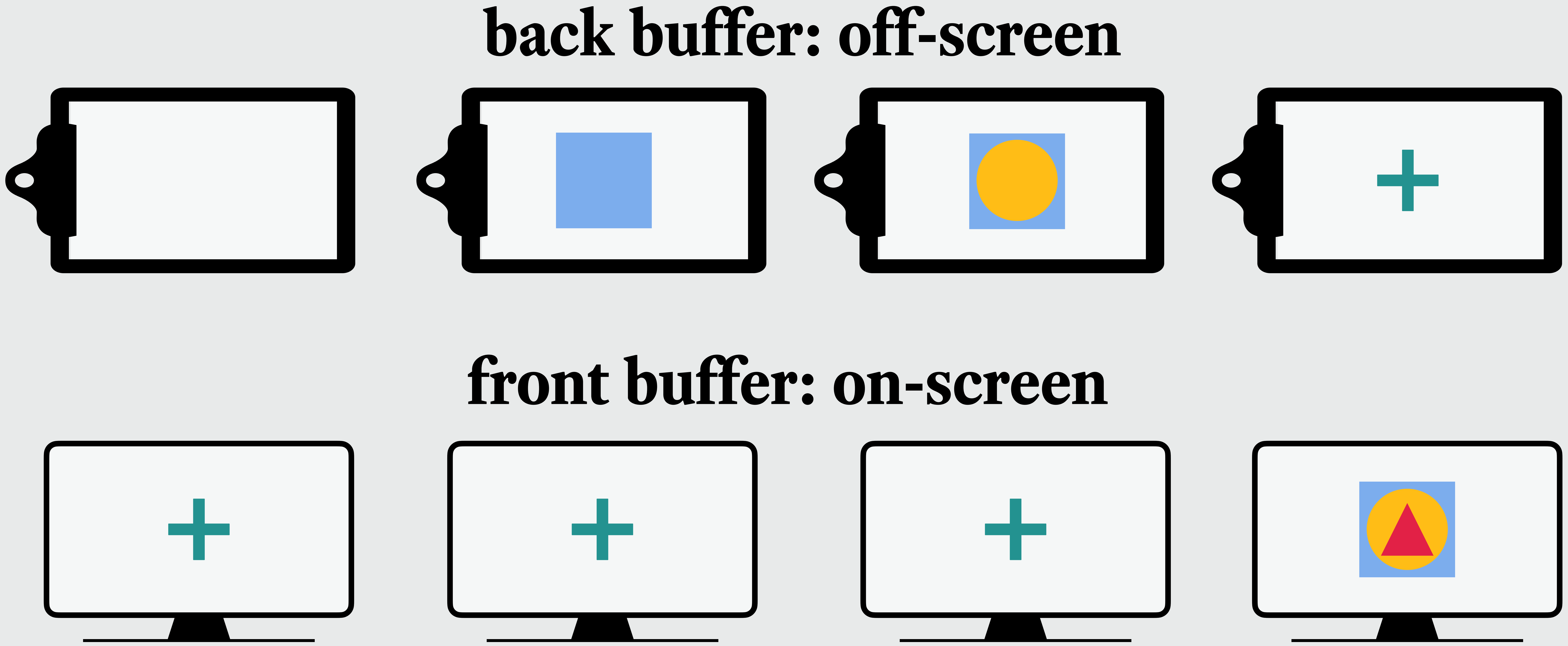
Let’s dig into it: https://github.com/barburevencu/PPE/blob/main/Week-2/Instructions.md

The first expyriment script

```python
from expyriment import design, control, stimuli

exp = design.Experiment(name="Circle")
control.initialize(exp)

fixation = stimuli.FixCross()
circle = stimuli.Circle(radius=50)

control.start(subject_id=1)

fixation.present(clear=True, update=True)
exp.clock.wait(1000)

circle.present(clear=True, update=True)
exp.keyboard.wait()

control.end()
```

stimulus.present()

```
square.present(clear=True, update=False)
circle.present(clear=False, update=False)
triangle.present(clear=False, update=True)
```



back buffer: off-screen

front buffer: on-screen

Solve Exercise 1

Exercise 1: Superimposed objects

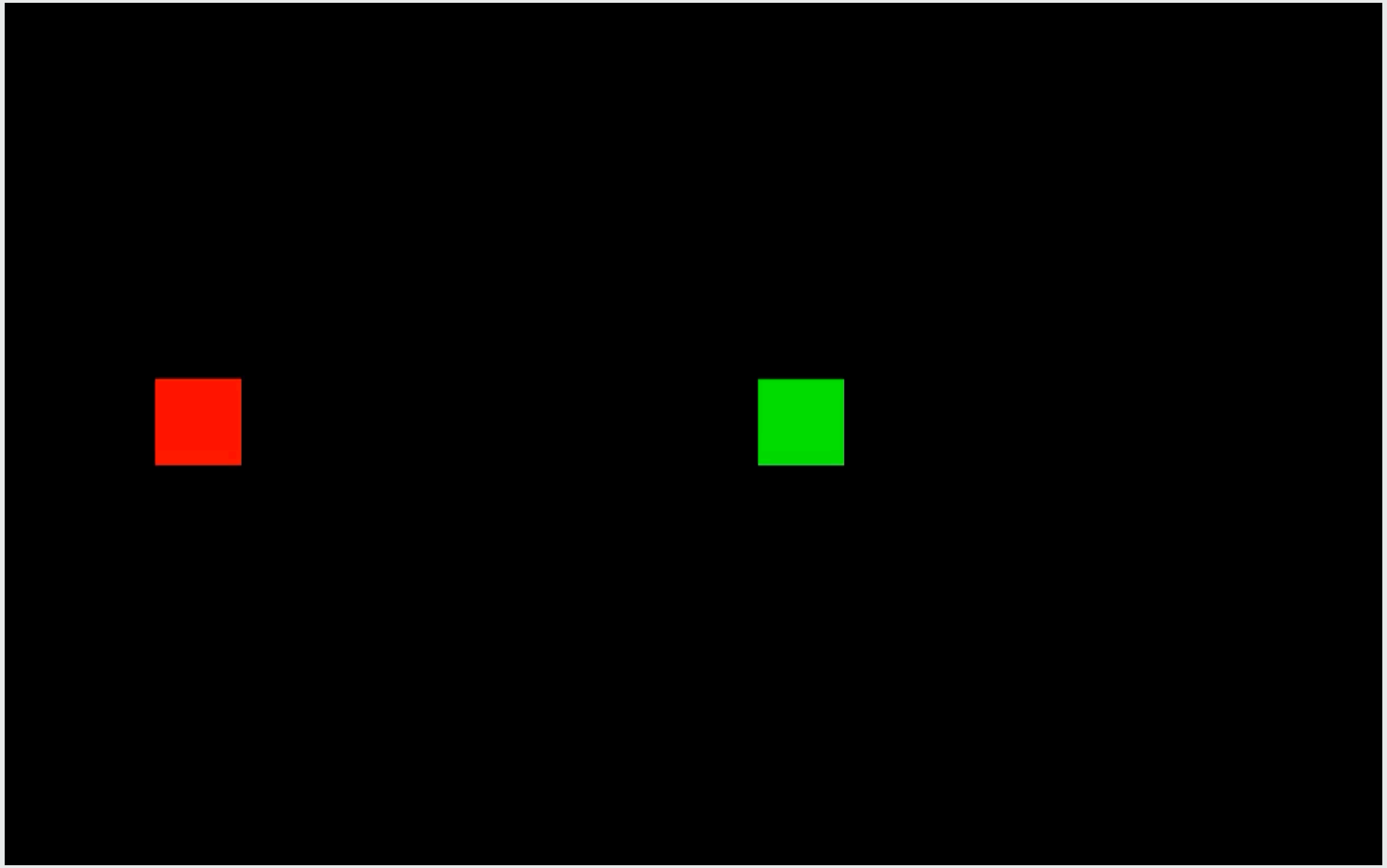
```
from expyriment import design, control, stimuli

...

square = stimuli.Rectangle(size=(50, 50), colour=(0, 0, 255))

...

square.present(clear=True, update=False)
fixation.present(clear=False, update=True)

exp.clock.wait(500)

square.present(clear=True, update=True)
```

Solve Exercise 2

Exercise 2: Two squares

```
from expyriment import design, control, stimuli

...

square_size = (50, 50)

left_square = stimuli.Rectangle(
     size=square_size,
     colour=(0, 0, 255),
     position=(-100, 0)
   )

...

left_square.present(clear=True, update=False)
right_square.present(clear=False, update=True)
```

Solve Exercise 3A



Exercise 3A: Michottean launching

```
# Distance to travel = Initial distance between objects
displacement_x = 400

# Set speed
step_size = 10 # pixels per update

# Move left square until collision
while right_square.position[0] - left_square.position[0] < square_length:
   left_square.move((step_size, 0)) # (move-x, move-y)
   # Don’t forget to update the screen!
   ...

# Move right square the same amount
while right_square.position[0] < displacement_x:
   right_square.move((step_size, 0))

# A better way (expyriment): r_square.distance(l_square) < square_length
```

Push your work to GitHub

Homework

Exercises 3B–E: Play around with different parameters to probe your causal perception

Exercise 3E: Launching function

Exercise 3F: Optional challenge

Exercises 4A–B: Shape, Text, and Line stimuli