

A reliable participant experience

Meeting 4 | Presentation, assets, browser assumptions, and pilot evidence

Bartosz Maćkiewicz

Programming Behavioral Experiments in jsPsych

Make the task readable and make predictable failures visible.

Artifact: an image-categorization pilot with controlled layout, explicit preload, a browser check, and a pilot log.

Retrieval: what does running once prove?

One successful run shows that one path worked in one environment.

It does not establish correct mappings, complete data, working failure paths, device compatibility, or research validity.

Name one additional test your project needs.

Instructions are part of the method

Tell participants:

- what to attend to;
- which response means what;
- whether speed matters and the deadline;
- what practice and completion look like;
- how to stop.

Use familiar language before technical terms.

```
<section class="card">  
  <h1>Shape categorization</h1>  
  <p>F = rounded; J = angular.</p>  
  <p>You have 3 seconds per shape.</p>  
</section>
```

The heading, mapping, and action should be easy to find.

```
.card {  
  max-width: 46rem;  
  padding: 1.5rem;  
  background: white;  
}
```

Keep spacing, stimulus size, and response reminders consistent. Avoid decorative changes that introduce condition differences.

Add the **card** class and improve one sentence.

Ask a partner to explain the mapping without your help. Test readability at the classroom display size.

The core styles are in `.shared/course.css`.

A file path is part of the experiment

Images are local assets in `materials/assets/`.

Check exact filename, case, and relative location.

A path that works in your folder may break when only one file is copied. Package and test the whole experiment.

Preload the actual item list

Use `images: SHAPE_STIMULI.map((item) => item.file)`. Set `continue_after_error: false` for a task that requires every image.

Dynamic paths need an explicit preload list. Test a missing file deliberately.

Preload both images, then run the response procedure.

Verify a rounded and an angular shape, their keys, and rows. Temporarily break a filename; confirm the error path; then restore it and rerun.

A controlled image trial

```
const trial = {  
  type: jsPsychImageKeyboardResponse,  
  stimulus: jsPsych.timelineVariable("file"),  
  stimulus_height: 220,  
  choices: ["f", "j"],  
  trial_duration: 3000  
};
```

Displayed size is a design choice. Pixels are not visual angle.

Browser checks make assumptions explicit

```
const check = {  
  type: jsPsychBrowserCheck,  
  minimum_width: 800,  
  minimum_height: 600,  
  inclusion_function: (data) => !data.mobile  
};
```

A viewport check is not proof of a physical keyboard.

Reduce the browser viewport and inspect the resize message.

Restore it and verify continuation. Discuss what the mobile exclusion tests and what it cannot guarantee.

Choose thresholds from task requirements, not habit.

Preload helps avoid missing or delayed assets.

It does not remove refresh-rate, keyboard, browser, or operating-system differences.

Do not infer millisecond-accurate physical presentation from a number in your source file alone.

A requested 200 ms duration is not a measurement of physical display onset or duration. The browser, display, and device also matter.

- Keep text legible and focus visible.
- Provide a non-color alternative when color is not essential.
- Explain required input devices.
- Keep camera activities optional.

If an adaptation changes the measure, document the change.

1. Preload the complete item list.
2. Control stimulus size and randomize order.
3. Check scoring and clear summaries.
4. Run a failure test and a fresh-folder test.

Record what actually happened in the pilot log.

Pilot log: expected versus observed

Case	Expected	Evidence
Valid response	Correct item/key row	Export
Timeout	Missing response, false correctness	Export
Missing image	Helpful stop	Screen/error
Small viewport	Resize route	Screen
Download	File opens with final row	CSV/JSON

Do not confuse a pilot with a result

A pilot tests instructions, flow, loading, and data.

It can reveal design problems. It does not need to produce a significant effect. Label synthetic/self-test data clearly.

Use eight minutes per project:

- explain the question;
- run one critical path;
- inspect one row;
- agree the most important next fix.

Peer feedback is formative, not a grade.

Show one failure you deliberately tested and its evidence.

Before Meeting 5: submit your pilot, list remaining bugs, and read the deployment guide. A webcam is optional.