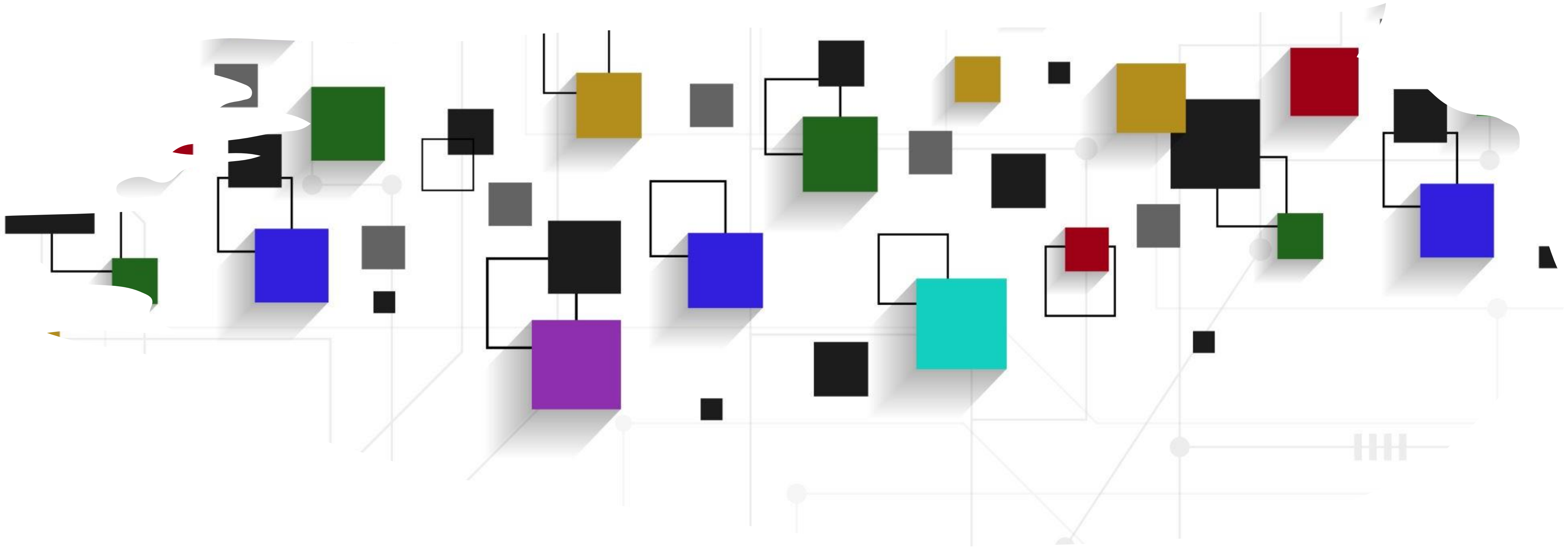


CogLab: jsPsych conditionals

WEEK 4

logistics: group project

- milestone #2: proposal
  - feedback returned
  - please stop by office hours / email us for other times!
- milestone #3: design draft
  - link to ONE github repository with preliminary code
  - make a list of stimuli files, plugins, etc. your experiment will need
  - the more you do now, the more we can help!

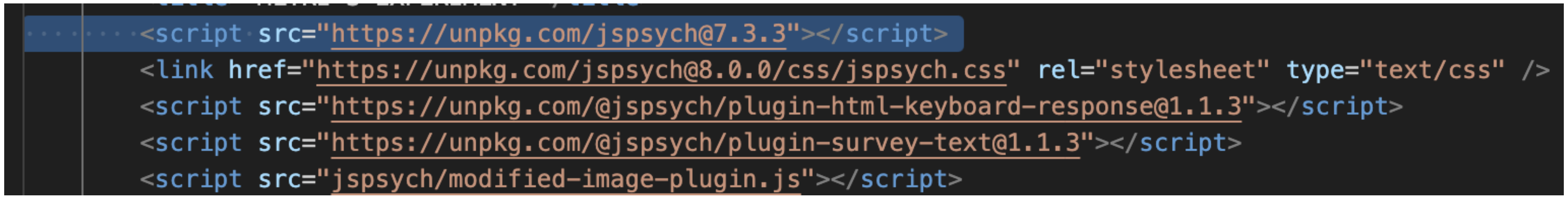
recap

- what we covered:
  - importing stimuli into jsPsych
  - repeating sequence of events for different items
- your to-dos were:
  - *prep:* conditional timelines and providing feedback
  - *prep:* design draft (project milestone #4)

going back to our experiment

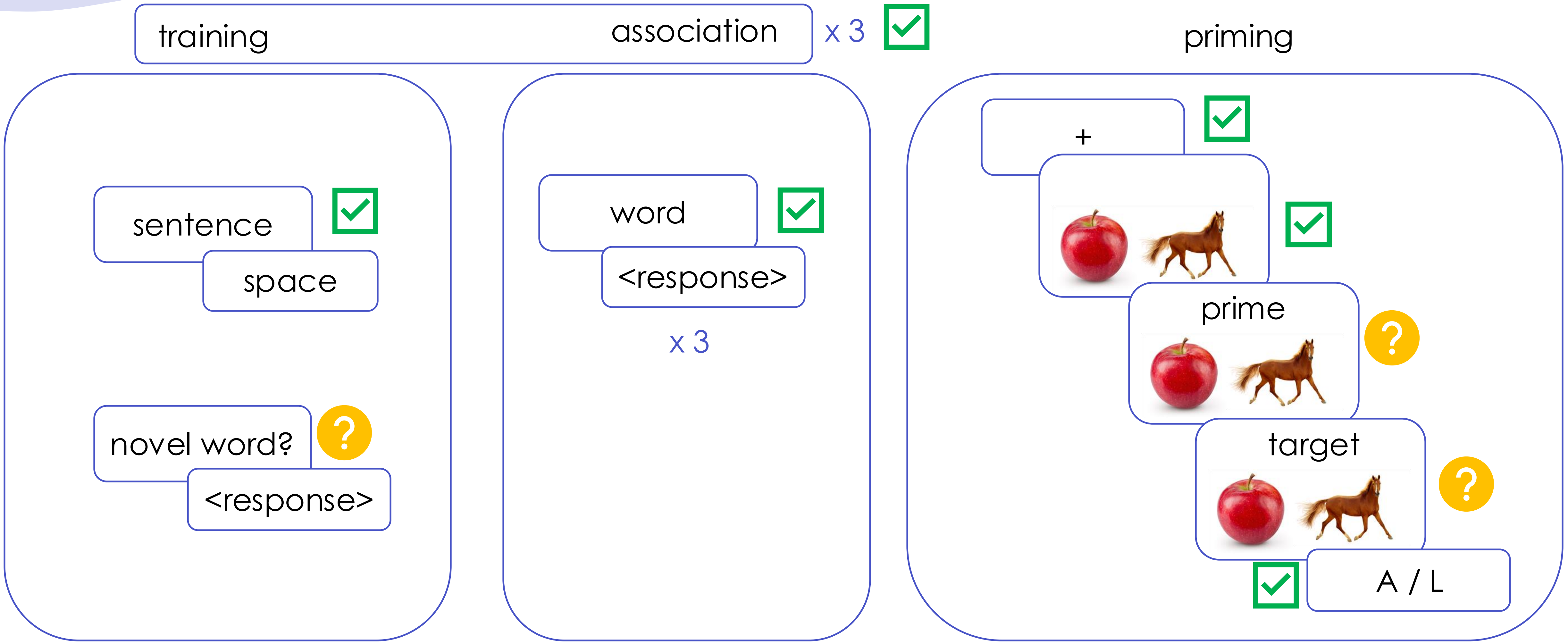
- open Visual Studio Xcode and open the jsPsych experiment you created last week
- also open the index.html file in your browser to remind yourself of what we did!

fixing the association procedure



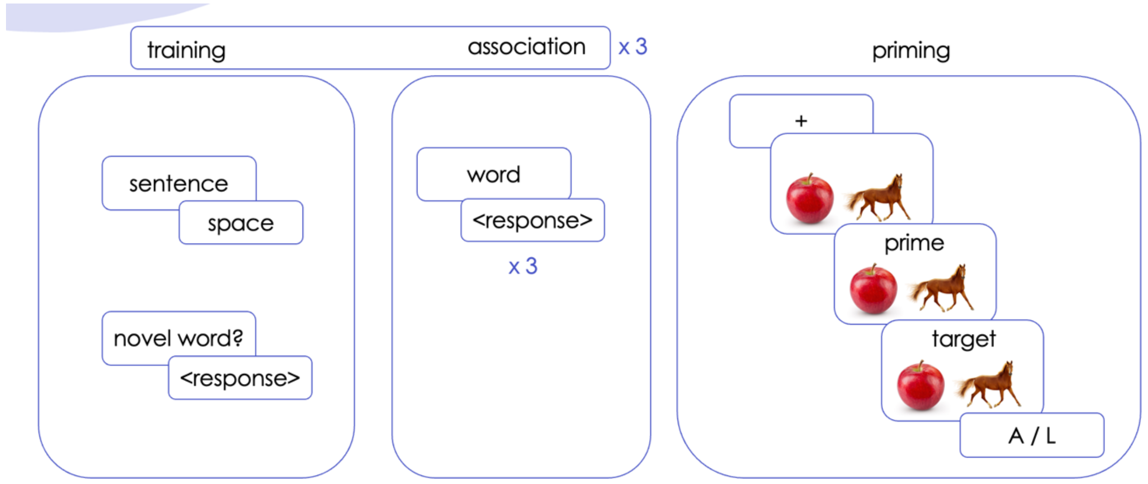
```
    <script src="https://unpkg.com/jspsych@7.3.3"></script>
    <link href="https://unpkg.com/jspsych@8.0.0/css/jspsych.css" rel="stylesheet" type="text/css" />
    <script src="https://unpkg.com/@jspsych/plugin-html-keyboard-response@1.1.3"></script>
    <script src="https://unpkg.com/@jspsych/plugin-survey-text@1.1.3"></script>
    <script src="jspsych/modified-image-plugin.js"></script>
```

experiment recap



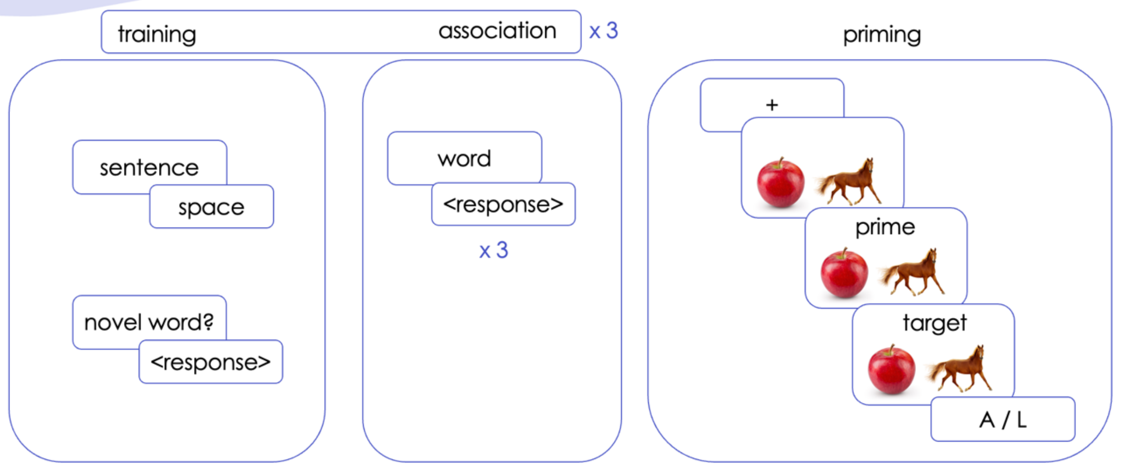
outstanding issues

- fixing position & style of prime/target words
- adding instruction screens
- attention checks
- feedback
- recording data



today’s agenda: outstanding issues

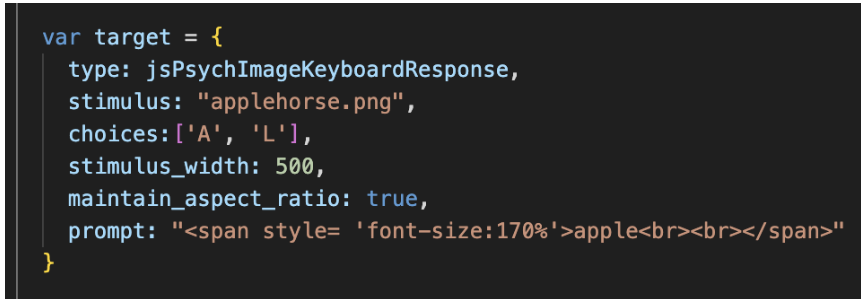
- fixing position & style of prime/target words
- adding instruction screens
- attention checks
- feedback
- recording data



prime/target presentation

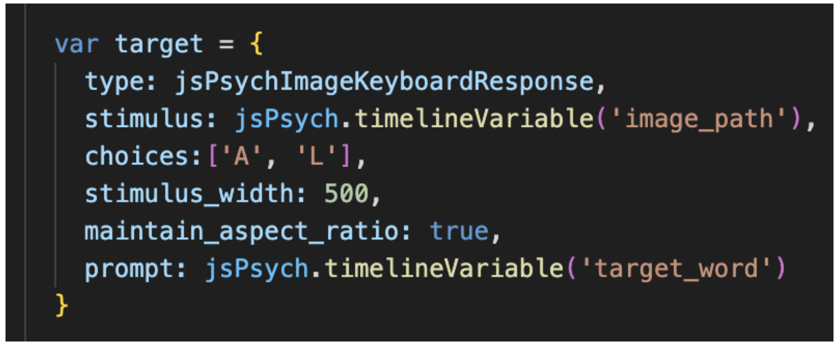
- previously, we had added breaks (<br>) and styling (<span>) to the prompt
- we need to add this back in a way that is compatible with the value returned by the jsPsych.timelineVariable() function

before



```
var target = {
  type: jsPsychImageKeyboardResponse,
  stimulus: "applehorse.png",
  choices:['A', 'L'],
  stimulus_width: 500,
  maintain_aspect_ratio: true,
  prompt: "<span style= 'font-size:170%'>apple<br><br></span>"
}
```

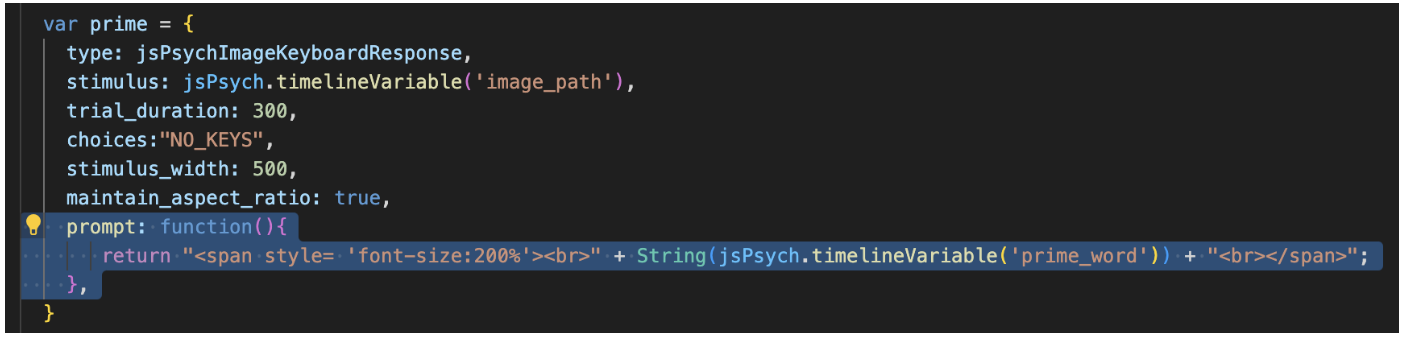
after



```
var target = {
  type: jsPsychImageKeyboardResponse,
  stimulus: jsPsych.timelineVariable('image_path'),
  choices:['A', 'L'],
  stimulus_width: 500,
  maintain_aspect_ratio: true,
  prompt: jsPsych.timelineVariable('target_word')
}
```

modifying prime plugin

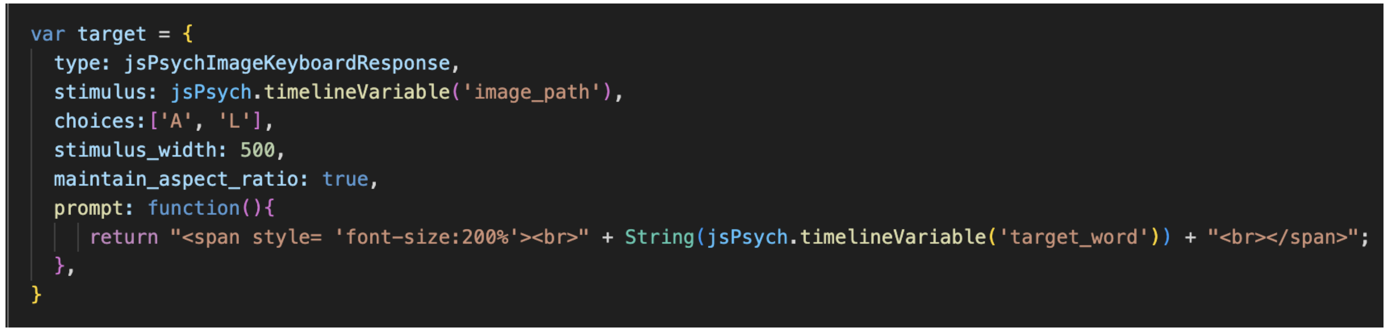
- instead of directly assigning prompt the value returned by the timelineVariable, we instead assign it the value from a function that returns a string of formatted primes



```
var prime = {
  type: jsPsychImageKeyboardResponse,
  stimulus: jsPsych.timelineVariable('image_path'),
  trial_duration: 300,
  choices:"NO_KEYS",
  stimulus_width: 500,
  maintain_aspect_ratio: true,
  prompt: function(){
    return "<span style= 'font-size:200%'><br>" + String(jsPsych.timelineVariable('prime_word')) + "<br></span>";
  },
}
```

modifying target plugin

- repeat for target plugin
- save and reload



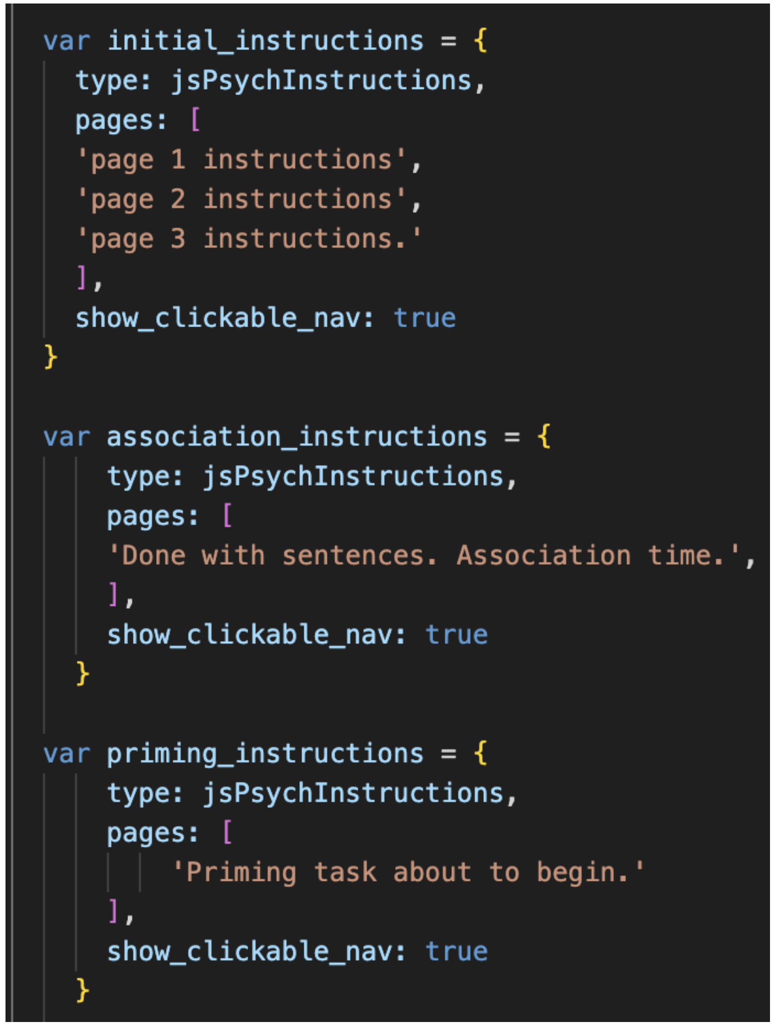
```
var target = {
  type: jsPsychImageKeyboardResponse,
  stimulus: jsPsych.timelineVariable('image_path'),
  choices:['A', 'L'],
  stimulus_width: 500,
  maintain_aspect_ratio: true,
  prompt: function(){
    return "<span style= 'font-size:200%'><br>" + String(jsPsych.timelineVariable('target_word')) + "<br></span>";
  },
}
```

outstanding issues / today's agenda

- fixing position & style of prime/target words
- adding instruction screens
- attention checks
- feedback
- recording data

adding instruction screens

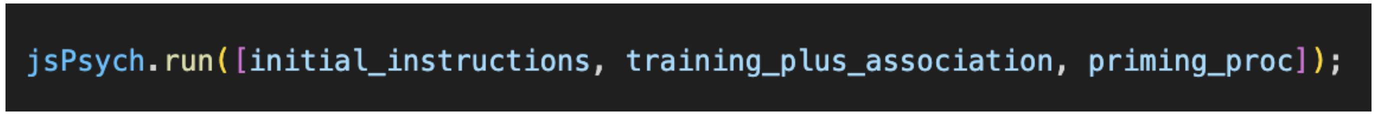
- adding instructions is a crucial part of guiding the participant through your experiment
- load the instructions plugin
- add three instruction trials
  - at the start of the experiment
  - before association
  - before priming



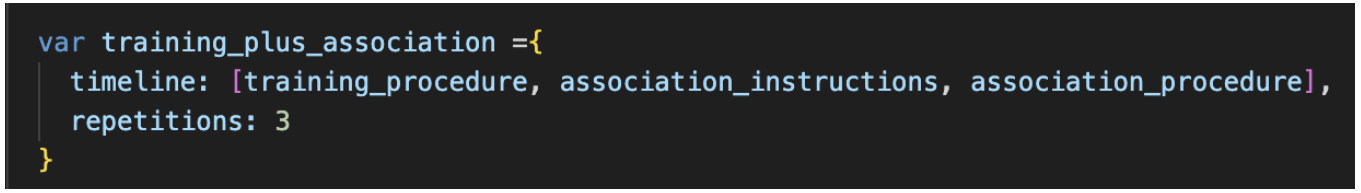
```
var initial_instructions = {
  type: jsPsychInstructions,
  pages: [
  'page 1 instructions',
  'page 2 instructions',
  'page 3 instructions.'
  ],
  show_clickable_nav: true
}

var association_instructions = {
    type: jsPsychInstructions,
    pages: [
    'Done with sentences. Association time.',
    ],
    show_clickable_nav: true
  }

var priming_instructions = {
    type: jsPsychInstructions,
    pages: [
        'Priming task about to begin.'
    ],
    show_clickable_nav: true
  }
```

incorporating instruction trials

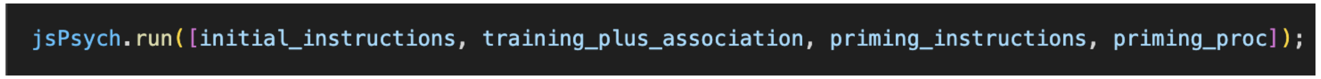
- initial_instructions can directly be part of the jsPsych.run() call
- association_instructions need to be displayed at the end of each sentence block
- priming_instructions need to be displayed at the end of the training_plus_association sequence
- save and reload



```
jsPsych.run([initial_instructions, training_plus_association, priming_proc]);
```



```
var training_plus_association ={
  timeline: [training_procedure, association_instructions, association_procedure],
  repetitions: 3
}
```



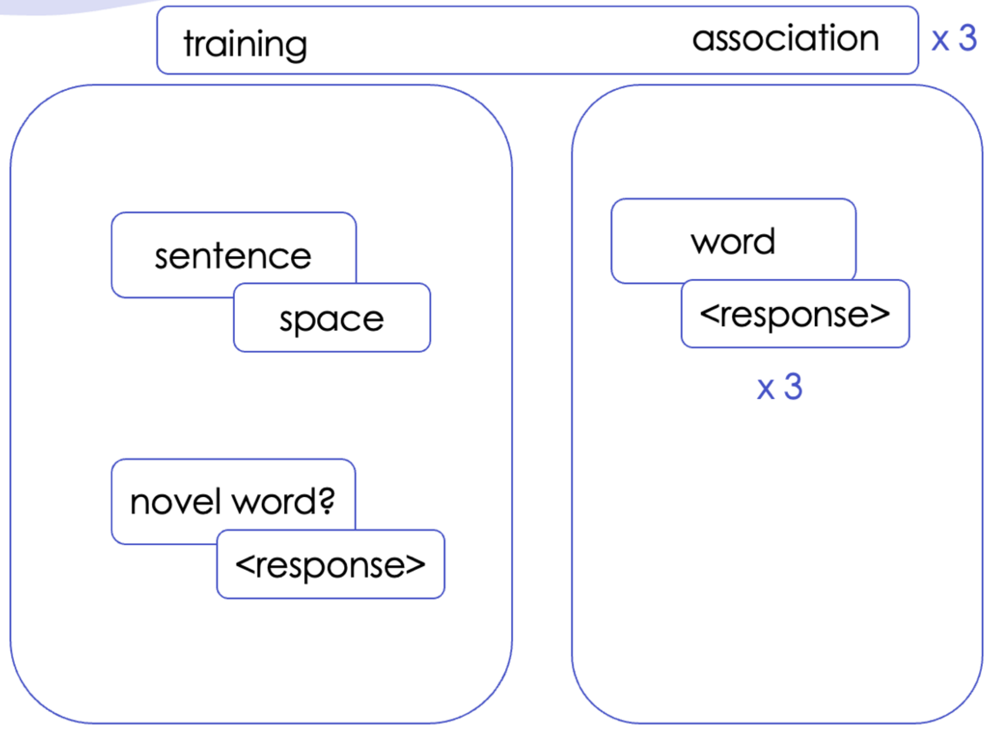
```
jsPsych.run([initial_instructions, training_plus_association, priming_instructions, priming_proc]);
```

outstanding issues / today’s agenda

- fixing position & style of prime/target words
- adding instruction screens
- attention checks
- feedback
- recording data

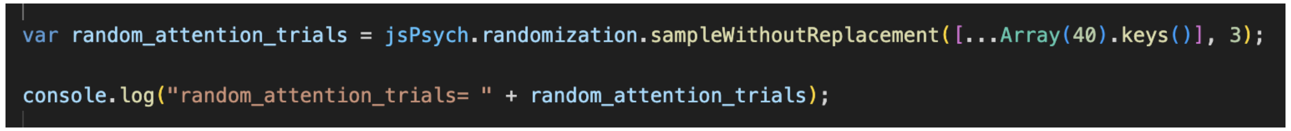
logic of attention check

- we want the attention check to appear at random points during the experiment
  - "each block contained attention check questions and a free association task"
  - "attention checks followed three randomly selected Training sentences"
- this means attention checks are conditional

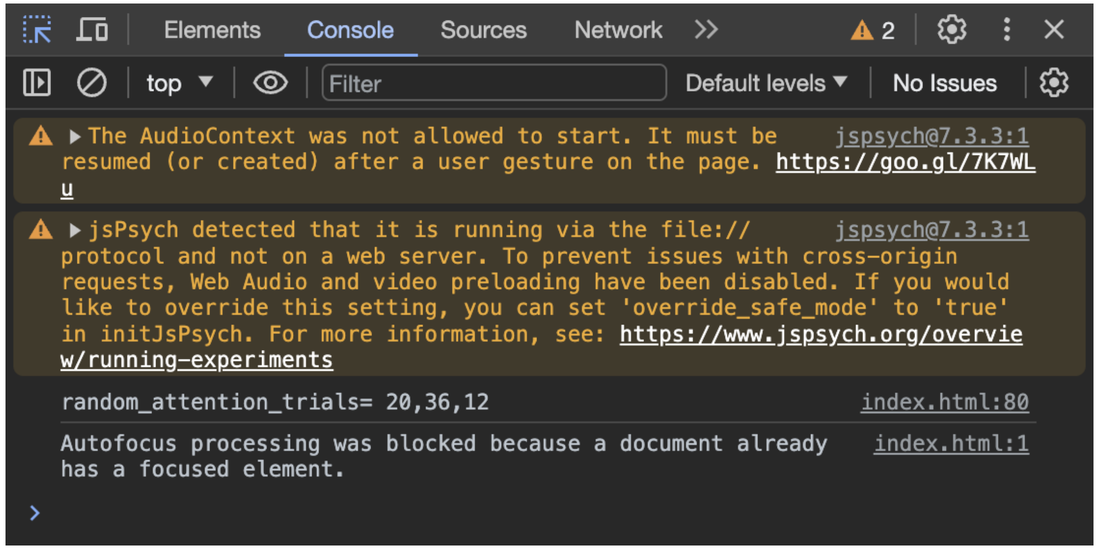


choosing random trials

- we need to randomly select any three trials from all the sentence trials
- we have 40 sentences, and counts start at 0 and end at 39 in JavaScript
- we define a variable that uses the jsPsych.randomization.sampleWithoutReplacement() function to get 3 random numbers between 0 and 40
- we also print the values to the console using console.log() so we can look at them in the inspector!
- save and reload page, and open the inspector via Command + Option + I



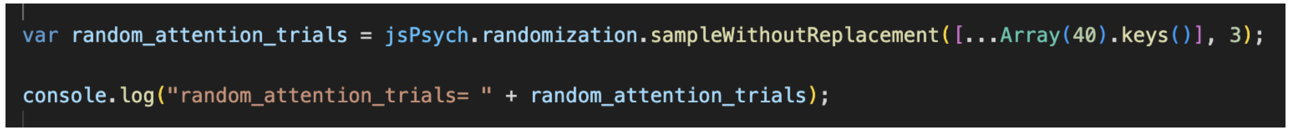
```
var random_attention_trials = jsPsych.randomization.sampleWithoutReplacement([...Array(40).keys()], 3);

console.log("random_attention_trials= " + random_attention_trials);
```



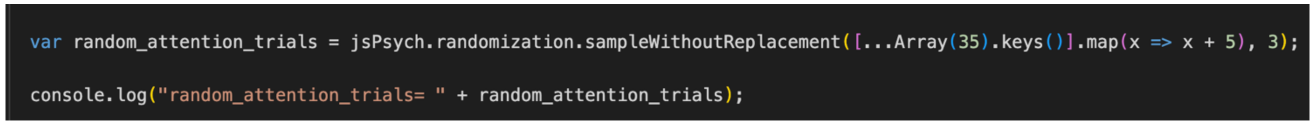
```
▶ The AudioContext was not allowed to start. It must be      jspsych@7.3.3:1
resumed (or created) after a user gesture on the page. https://goo.gl/7K7WL
u
▶ jsPsych detected that it is running via the file://        jspsych@7.3.3:1
protocol and not on a web server. To prevent issues with cross-origin
requests, Web Audio and video preloading have been disabled. If you would
like to override this setting, you can set 'override_safe_mode' to 'true'
in initJsPsych. For more information, see: https://www.jspsych.org/overvie
w/running-experiments
random_attention_trials= 20,36,12                            index.html:80
Autofocus processing was blocked because a document already   index.html:1
has a focused element.
>
```

range of random trials

- currently, the attention check could happen even on the first sentence trial, which would be strange
- we can restrict this by modifying our code slightly
- we sample from 0 to 34 and add 5 to the random sample
  - minimum / maximum?
- save and reload, open inspector



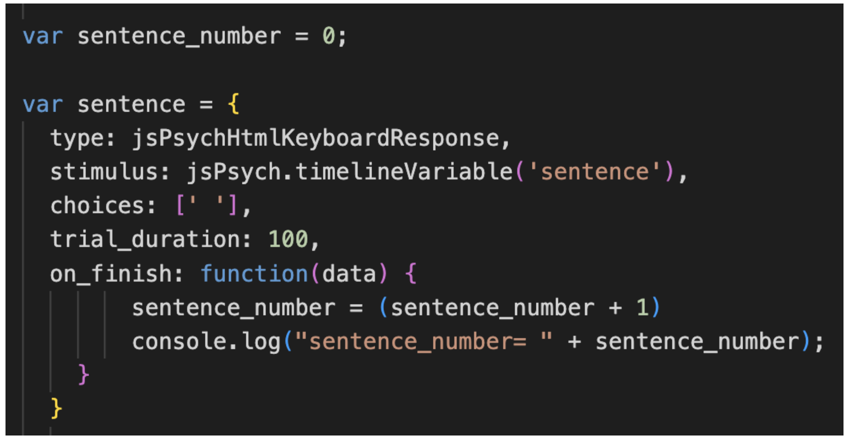
```
var random_attention_trials = jsPsych.randomization.sampleWithoutReplacement([...Array(40).keys()], 3);

console.log("random_attention_trials= " + random_attention_trials);
```



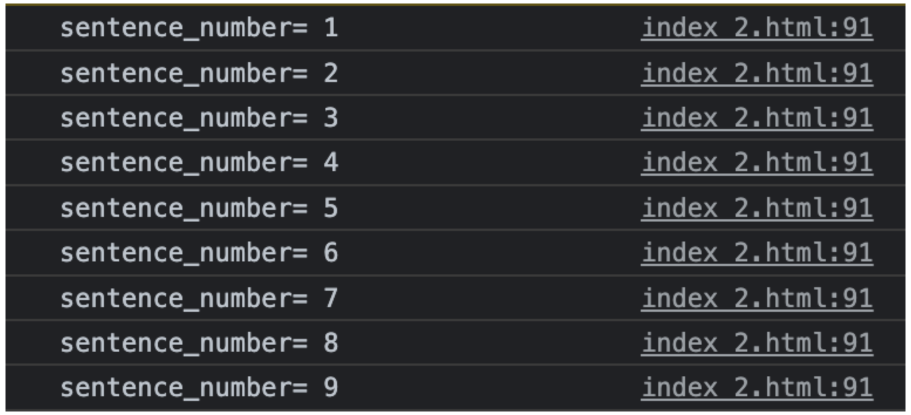
```
var random_attention_trials = jsPsych.randomization.sampleWithoutReplacement([...Array(35).keys()].map(x => x + 5), 3);

console.log("random_attention_trials= " + random_attention_trials);
```

keeping track of the sentence number

- once we have the random trials chosen, we need to have an attention check at those times
- so, we need to keep a count of sentences
- for this, we modify the sentence plugin trial to track the trials where sentences are presented using the on_finish parameter
- save and reload
- open the inspector



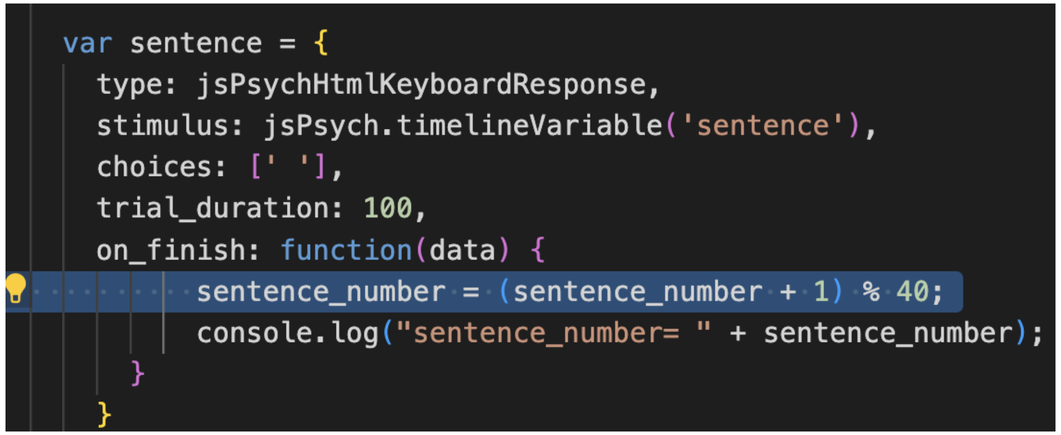
```
var sentence_number = 0;

var sentence = {
  type: jsPsychHtmlKeyboardResponse,
  stimulus: jsPsych.timelineVariable('sentence'),
  choices: [' '],
  trial_duration: 100,
  on_finish: function(data) {
        sentence_number = (sentence_number + 1)
        console.log("sentence_number= " + sentence_number);
    }
  }
```



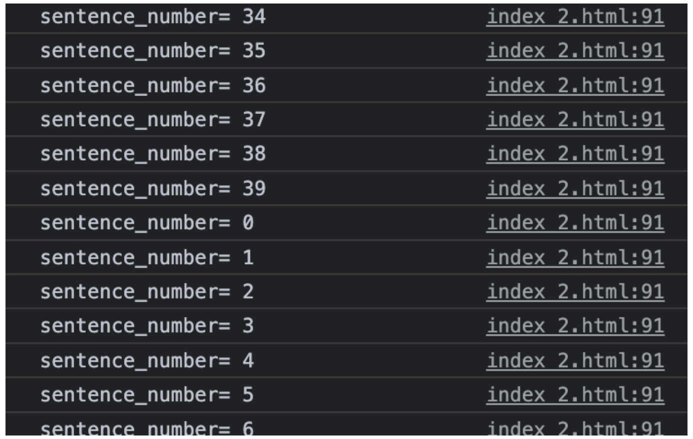
```
sentence_number= 1                         index_2.html:91
sentence_number= 2                         index_2.html:91
sentence_number= 3                         index_2.html:91
sentence_number= 4                         index_2.html:91
sentence_number= 5                         index_2.html:91
sentence_number= 6                         index_2.html:91
sentence_number= 7                         index_2.html:91
sentence_number= 8                         index_2.html:91
sentence_number= 9                         index_2.html:91
```

restricting the trial number range

- random_attention_trials will always be within 5 and 39 by design, but our sentence_number keeps increasing across the 3 blocks
- solution: we divide the index by 40 and keep the remainder, using the % operator
- save and reload



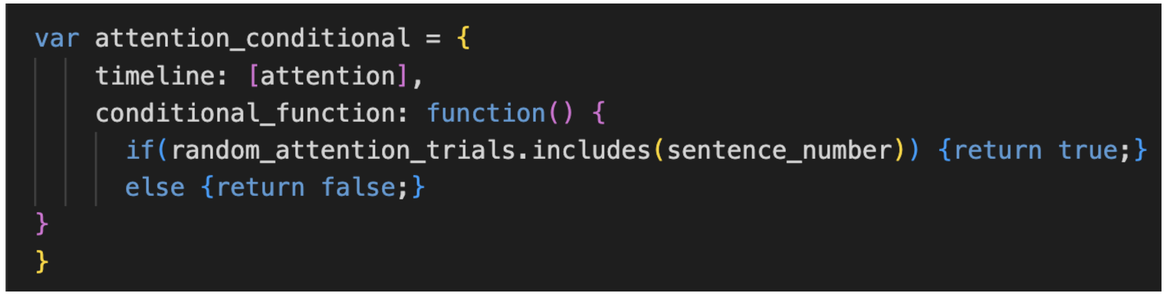
```
var sentence = {
  type: jsPsychHtmlKeyboardResponse,
  stimulus: jsPsych.timelineVariable('sentence'),
  choices: [' '],
  trial_duration: 100,
  on_finish: function(data) {
    sentence_number = (sentence_number + 1) % 40;
    console.log("sentence_number= " + sentence_number);
  }
}
```



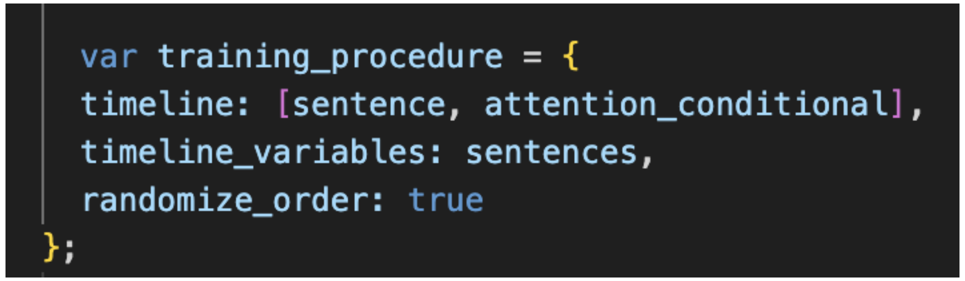
```
sentence_number= 34                              index 2.html:91
sentence_number= 35                              index 2.html:91
sentence_number= 36                              index 2.html:91
sentence_number= 37                              index 2.html:91
sentence_number= 38                              index 2.html:91
sentence_number= 39                              index 2.html:91
sentence_number= 0                               index 2.html:91
sentence_number= 1                               index 2.html:91
sentence_number= 2                               index 2.html:91
sentence_number= 3                               index 2.html:91
sentence_number= 4                               index 2.html:91
sentence_number= 5                               index 2.html:91
sentence_number= 6                               index 2.html:91
```

defining a conditional timeline

- we can now define a conditional timeline and use the sentence_number and the random_attention_trials to only display the attention trial if the sentence_number is in the random_attention_trials
- add attention_conditional to the training_procedure
- save and reload



```
var attention_conditional = {
    timeline: [attention],
    conditional_function: function() {
        if(random_attention_trials.includes(sentence_number)) {return true;}
        else {return false;}
    }
}
```



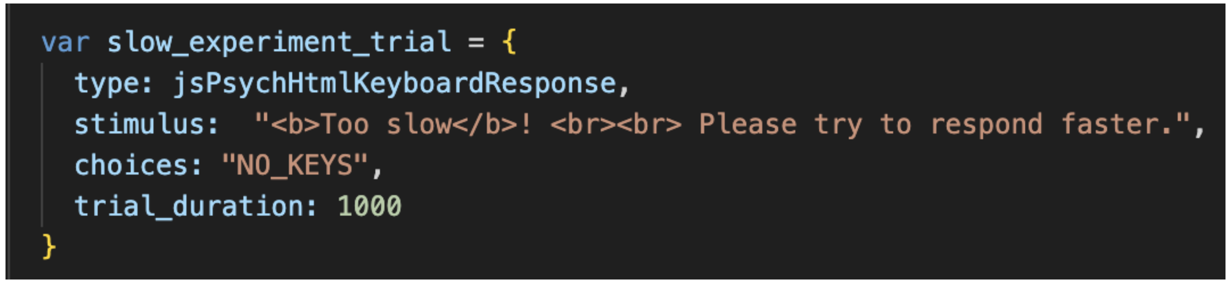
```
    var training_procedure = {
    timeline: [sentence, attention_conditional],
    timeline_variables: sentences,
    randomize_order: true
};
```

outstanding issues / today's agenda

- fixing position & style of prime/target words
- adding instruction screens
- attention checks
- feedback
- recording data

creating the feedback screen

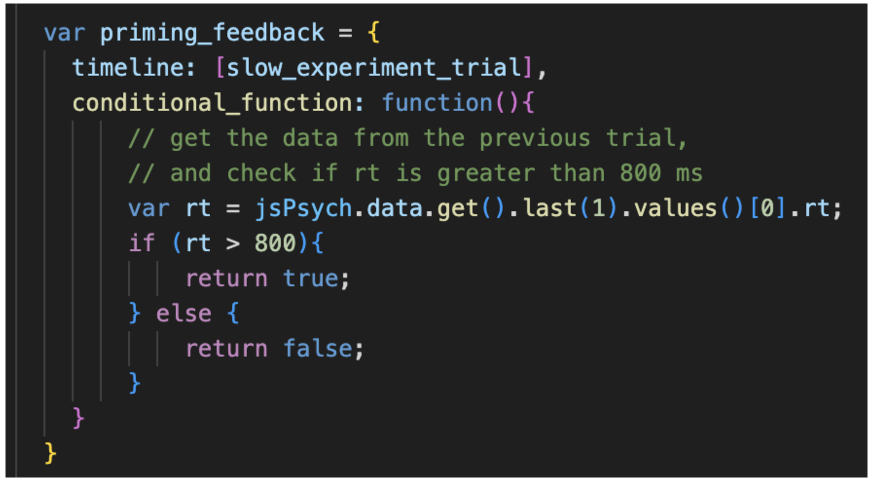
- we first define a slow_experiment_trial that displays feedback to the participants



```
var slow_experiment_trial = {
  type: jsPsychHtmlKeyboardResponse,
  stimulus:  "<b>Too slow</b>! <br><br> Please try to respond faster.",
  choices: "NO_KEYS",
  trial_duration: 1000
}
```

providing feedback on priming trials

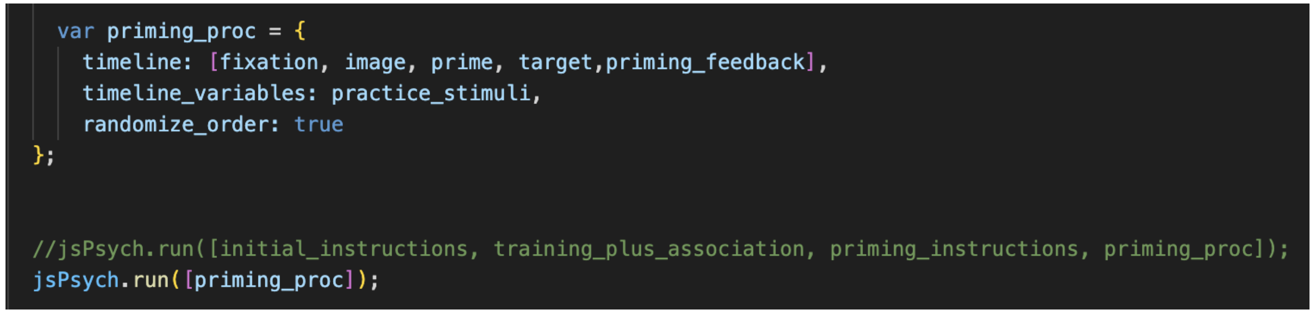
- if we want to provide feedback, we have to retrieve the data provided by the participant on the specific trial
- we define a conditional priming_feedback trial to only run the slow_experiment_trial plugin if RT on the last trial is > 800 ms



```
var priming_feedback = {
  timeline: [slow_experiment_trial],
  conditional_function: function(){
      // get the data from the previous trial,
      // and check if rt is greater than 800 ms
      var rt = jsPsych.data.get().last(1).values()[0].rt;
      if (rt > 800){
          return true;
      } else {
          return false;
      }
  }
}
```

add feedback to priming procedure

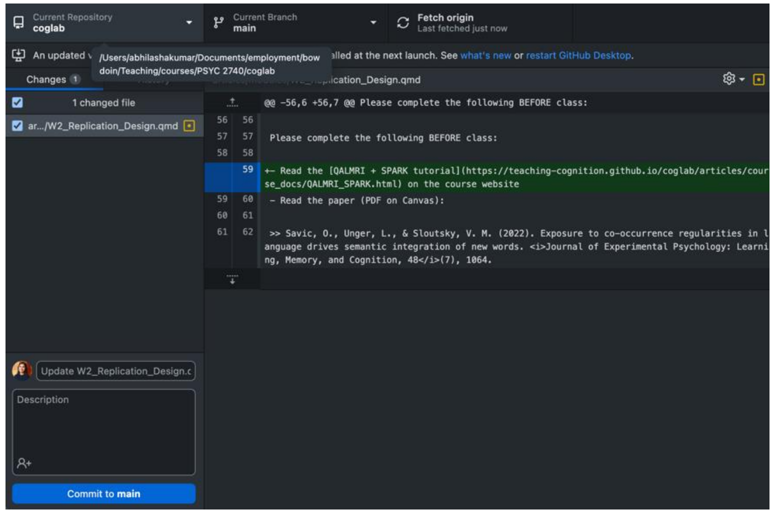
- we add priming_feedback to our priming procedure
- we modify the jsPsych.run() sequence to test the new priming procedure
- save and reload



```
  var priming_proc = {
    timeline: [fixation, image, prime, target,priming_feedback],
    timeline_variables: practice_stimuli,
    randomize_order: true
};


//jsPsych.run([initial_instructions, training_plus_association, priming_instructions, priming_proc]);
jsPsych.run([priming_proc]);
```

saving your progress so far…

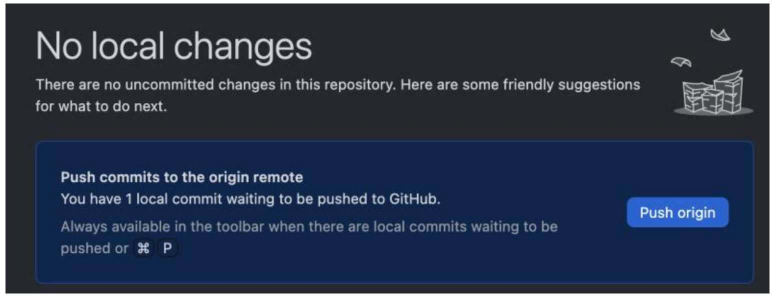
- save your index.html file
- open GitHub Desktop
- review changes, commit, and push
- check if changes have reflected online!



An updated v /Users/abhilashakumar/Documents/employment/bowdoin/Teaching/courses/PSYC 2740/coglab alled at the next launch. See what's new or restart GitHub Desktop.

ication_Design.qmd

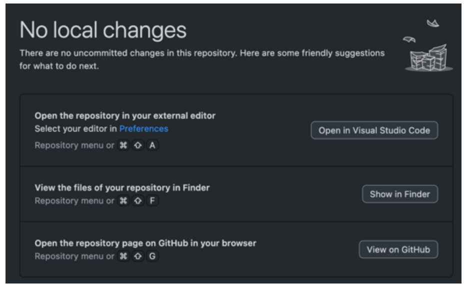
```
@@ -56,6 +56,7 @@ Please complete the following BEFORE class:

 Please complete the following BEFORE class:

+- Read the [QALMRI + SPARK tutorial](https://teaching-cognition.github.io/coglab/articles/course_docs/QALMRI_SPARK.html) on the course website
 - Read the paper (PDF on Canvas):

 >> Savic, O., Unger, L., & Sloutsky, V. M. (2022). Exposure to co-occurrence regularities in language drives semantic integration of new words. <i>Journal of Experimental Psychology: Learning, Memory, and Cognition, 48</i>(7), 1064.
```



No local changes

There are no uncommitted changes in this repository. Here are some friendly suggestions for what to do next.

Push commits to the origin remote
You have 1 local commit waiting to be pushed to GitHub.
Always available in the toolbar when there are local commits waiting to be pushed or ⌘ P



No local changes

There are no uncommitted changes in this repository. Here are some friendly suggestions for what to do next.

Open the repository in your external editor
Select your editor in Preferences
Repository menu or ⌘ ⇧ A

View the files of your repository in Finder
Repository menu or ⌘ ⇧ F

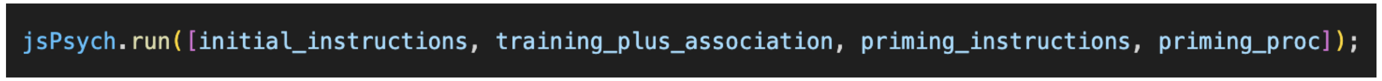
Open the repository page on GitHub in your browser
Repository menu or ⌘ ⇧ G

outstanding issues / today's agenda

- fixing position & style of prime/target words
- adding instruction screens
- attention checks
- feedback
- recording data

complete experiment procedure

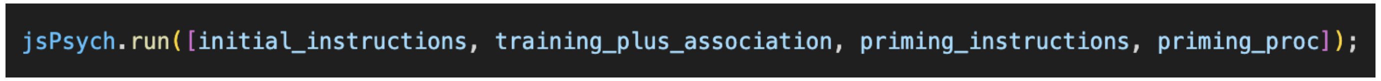
- initial instructions
- training plus association
  - sentences
  - some attention trials
  - association instructions
  - association trials
- priming procedure
  - priming instructions
  - fixation
  - image
  - prime
  - target
  - feedback



```
jsPsych.run([initial_instructions, training_plus_association, priming_instructions, priming_proc]);
```

HW: what data do we need for each plugin?

- initial instructions
- training plus association
    - sentences
    - some attention trials
    - association instructions
    - association trials
- priming procedure
    - priming instructions
    - fixation
    - image
    - prime
    - target
    - feedback



```
jsPsych.run([initial_instructions, training_plus_association, priming_instructions, priming_proc]);
```

HW: what does jsPsych automatically record?

- head over to the plugins page
- navigate to the pages for the plugins we are using
- look at the Data Generated sub-heading
- make note of what is being recorded and what else may be needed

next class

- **before** class
  - *prep:* class HW (data being recorded + data needed)
  - *prep:* read the online documentation on data storage
  - *apply:* project milestone #3 (design draft)
- **during** class
  - recording data
  - going online!