RECOGNITION MEMORY EXPERIMENT FRAMEWORK

DESIGNERS:
M. RABE MMRABE@UVIC.CA
DR. S. LINDSAY SLINDSAY@UVIC.CA

DEVELOPER:
A. RICHARDSON
RICHARDSON.ASHLIN@GMAIL.COM

INSTITUTION:
UNIVERSITY OF VICTORIA

CONTENTS

Date: May 23, 2017.

OVERVIEW

0.1. **Audience.** The intended audience for this document is a researcher with some programming experience.

0.2. **Description.** The **Recognition Memory Experiment Framework** is an online utility for parametric generation of **Recognition Memory experiments** to support researchers at the University of Victoria. The software is intended to be web based, self contained yet comprehensive, and reasonably flexible.

The software is a foundation upon which to develop and deploy interactive surveys/questionnaires as an essential component of Recognition Memory experiment methodologies, and is comprised of two aspects:

(1) The **researcher facing** portion, which is a simplified API-style programming interface, that specifies sequencing of both:

- online interactive visual elements, and

- requests to the participant for feedback.

(2) The **participant facing** portion consists of

- an experimental survey (based on JavaScript/HTML5) which runs client-side in the participant's web browser, and is specified by the researcher in terms of:

    – the different stimuli or other interactive visual elements revealed to the participant, and

    – the possible responses/feedback requested of the partcipant, and

- a server-side program (written in Python) running on the web server administrated by the researcher, that receives user responses, timing, and other information sent back to the researcher by the JavaScript/HTML5 program.

0.3. **Requirements for using the software.**

0.3.1. *Server-side.*

- Host:

    – An ordinary web server with Python/CGI enabled, is required.

    – Note: the system was tested with server: Apache/2.2.23 (Unix).

0.3.2. *Client-side.*

- For experiment participants:

    – A modern web browser (Firefox, Google Chrome, or Safari) on a desktop computer is required.
        * The system was tested with Chrome v. 57, Safari 10.1, and Firefox 53.0 (64 bit).

- For researchers and experiment administrators:

    – A text editor is required to edit experiment script files.

    – Limited technical knowledge about JavaScript is required to edit or modify experiments.

    – An FTP program is required for uploading experiment scripts (and downloading response data).
        * A free/open implementation is FileZilla, available at https://filezilla-project.org/

1. The System

1.1. **Installation.** The system may be installed on a web server by downloading:

- https://github.com/ashlinrichardson/m3m0ry/archive/master.zip

and extracting the ZIP file to an administrator-determined folder/location on the server. Typically this might involve downloading

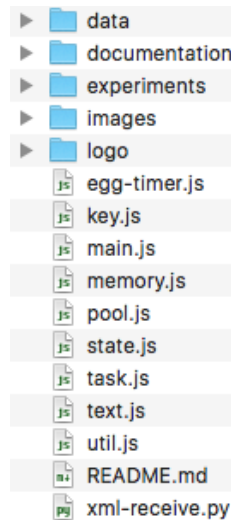1.2. **Project Structure.** The system has the following directory structure as in Fig. (1.1).

Figure 1.1.

In Fig. (1.1) above:

- **data**/ should, once survey(s) have been successfully completed, contain CSV data files representing the user experience:

  - If all goes well, an additional data file should automagically appear in the **data**/ folder, for any given survey/experiment that is successfully completed.

  - Upon completion of a survey/experiment, the client-side JavaScript code submits (via util.js::xml_send()) a CSV data file to the web server, which receives the data using CGI/Python (via xml-receive.py).

  - The CSV file is saved with a name reflecting the date/time when the file was recorded, and also a randomly-generated string that is added to prevent file-naming "collisions".

- **documentation**/ is where this manual resides.

- **experiments**/
  - Contains a number of sub-folders, one for each of the included examples:
    * **delay**/
    * **feedback**/
    * **instructions**/
    * **study-phase**/
    * **test-phase**/
    * **my-experiment**/

  - Each of the above subfolder contains a file **memory.html**, which always has the contents:

```
6 lines (5 sloc)    70 Bytes

1    <html>
2    <body>
3    <script src="../../memory.js"></script>
4    </body>
5    </html>
```

FIGURE 1.2.

  Note: any experiment/survey project developed by the user must also:
    * reside in the **experiments**/ folder (as with the examples provided), and:
    * include a **memory.html** file, which should be the same as in Fig. (1.2).

- **images**/ contains image data used in experiments. To change image data used in experiments, the administrator should:

  - upload new image data into the **images**/ folder, ensuring that the image data is consistently named according to the same numbered format followed by the provided image data: **1.jpg, 2.jpg,** and so on.

2. SETUP

2.1. Accessing an Existing Survey. Supposing the project is uploaded to the main HTTP directory of a web server with URL **http://my-web-server.com**/, the survey in the folder **experiments/my-experiment**/ as represented by the JavaScript file **experiments/my-experiment/my-experiment.js** administered by the researcher/administrator, will be accessed by navigating in a web broswer to the following address:

- **http://my-web-server.com/experiments/my-experiment/memory.html**

Of course, for any given experiment, a researcher/administrator is advised to thoroughly test the survey, in a web browser, by accessing a URL like the one above.

2.2. **Creating A New Survey or Modifying An Existing One.** To create your own experiment we recommend, with respect to the **experiments/** folder on the local computer (which will later be uploaded to the web server e.g., via FTP):

(1) making a copy (with a different name) of
- the folder corresponding to the given example provided which is
  - most closely representative of the experiment which one would like to create.

    * For example, if we wanted to make simple modifications to the instructions-type example, we could make a copy of the **experiments/instructions/** folder, and rename that folder **experiment/test-experiment/**

    * If we wanted to develop a more detailed and realistic complete experiment, we might make a copy of the experiment folder **experiment/my-experiment/**, and rename the copied folder to e.g.: **experiment/test-experiment/**

(2) given the folder we created, e..g, **experiment/test-experiment/,** we should

- edit the file **my-experiment.js** within the **my-experiment/** folder, for example:

  - we could change the messages on any instructions slides;

  - adjust timing parameters (e.g. ISI: Inter-Stimulus-Interval) for certain components;

  - add additional word/image stimuli to an experiment (study/test phase components) with respect to one or more "stimulus pools";

  - add further tasks to the experiment (more study/test phases, instructions, delay activities, or extra feedback questions).

(3) **To deploy such an experiment on the web,** please make sure to upload your revised folder on the local computer (e.g., **experiment/test-experiment/**) to the web server.

- Please ensure that the revised folder is uploaded in the correct location (i.e., within the **experiment/** relative to the main project folder, as it appears on the web server), and that the revised folder includes both a **my-experiment.js** file and a **memory.html** file.

3. The Examples

Five example surveys are included:

(1) **instructions/**: involving display of instructions text or other directions/information;

(2) **delay/**: involving presenting the user with free-form or other response fields, to occupy the participant between different experiment/survey components;

(3) **feedback/**: involving presenting the user with various (multiple-choice) opportunities to reply with feedback/information;

(4) **study-phase/**: involving presenting the user with a variety of stimuli (usually accompanied by instructions that indicate to try to remember the stimuli presented);

(5) **test-phase/**: involving requesting feedback from the user, with respect to a sequence of stimuli information;

(6) **my-experiment/**: a more-detailed example, representing a typical possible experiment.

3.1. **experiments/instructions/my-experiment.js.** The file my-experiment.js below exemplifies the required format of a **my-experiment.js** file, which must contain a function called my_experiment, as indicated in line 2:

> **var my_experiment = function(){**

Please note the closing bracket for that function, on line 29. For those not familiar with JavaScript code, the text which appears within the marks: /* */ is a "comment", e.g., /* this is a comment: text within a program that does not represent instructions to be executed */

3.1.1. *Instructions Statements.* Note that, e.g. in line 5:

> **instructions('welcome to the recognition memory experiment framework\n\n\n\t* please press any key to continue')**

this indicates presenting the text within the single quotation-marks, to the user. Note: "/n" and "/t" represent "control characters" which affect the flow of text on the screen: "/n" is the newline-character which causes the rendering of text to proceed on the next available line, and "\t" is the tab-character, which indents text several spaces. The researcher can insert as many such statements as they like. Note some possible modifications to the "vanilla" instructions statement above, as follows:

3.1.2. *Instructions: Fixed Duration.* For displaying instructions for a fixed interval (in milliSeconds) we can insert a code block as follows, as in lines 17-19:

> **var x = instructions('this information will be displayed for 5 seconds')**
> **x.set_expiry(5000)**
> **x.key_expiry = false**

3.1.3. *Instructions: Fixed Duration or User Intervention.* For instructions that are shown until a key is pressed, but are shown for (at most) a given fixed interval (in milliSeconds) we can insert a code block as follows, as in lines 22-24:

> **var x = instructions('this information will be displayed for 5 seconds')**
> **x.set_expiry(5000)**
> **x.key_expiry = true**

3.1.4. *The file my-experiment.js.*

```
1  /* recognition memory experiment set-up */
2  var my_experiment = function(){
3
4    /* instruction slide */
5    instructions('welcome to the recognition memory experiment framework\n\n\n\t* please press any key to
         continue')
6
7    /* instruction slide */
8    instructions('here is what happens when you put in a lot of text— if you put in lots of text, it
         might wrap around the edge\n\n\n\t* please press any key to continue')
9
10   /* instruction slide */
11   instructions('this is an instructions slide\n\n\n\t* please press any key to continue')
12
13   /* instruction slide */
14   instructions('this is an instructions slide with extra line breaks:\nsingle line break:\ndouble line
         break:\n\ntriple line break:\n\n\n\t* please press any key to continue')
15
16   /* instruction slide — fixed duration */
17   var x = instructions('this instructions slide will display for 5 seconds:\n\n\n\t* if you press a key
         , it will do nothing')
18   x.set_expiry(5000)
19   x.key_expiry = false
20
21   /* instruction slide — fixed duration or user intervention */
22   var y = instructions('this instructions slide will display for up to 5 seconds:\n\n\n\t* if you press
         a key, the transition will happen before 5 seconds is up')
23   y.set_expiry(5000)
24   y.key_expiry = true
```

```
25
26   /* instruction slide */
27   instructions ('this is a normal instructions slide\n\n\n\t* please press any key to continue ')
28
29 }
```

3.2. experiments/delay/my-experiment.js. Although other manifestations could be possible for the delay task, the current implementation involves a free-form reponse section to involve the participant in an activity other than observing stimuli (an interval-type activity to be deployed between other experimental components, e.g., between a study-phase component and the subsequent test-phase component).

3.2.1. *Delay Task.* The basic syntax for specifying a "delay" task is like that for the "instructions" task:

> **delay('please enter the names of as many countries as you can think of, followed by the escape key\n\n\n\t* please press any key to continue')**

By default, the delay task will continue collecting user input, until the <escape> key is pressed.

3.2.2. *Delay Task: Fixed Interval.* It's also possible to specify a "delay" task that doesn't wait for an <escape> key to be pressed: this version waits for a specified interval (in mS) instead. To operate the delay with a fixed interval, simply add a time parameter (in mS) to the invocation:

> **delay('please enter the names of as many countries as you can think of, in 5 seconds\n\n\n\t* please press any key to continue', 5000)**

where 5000 mS is five seconds.

3.2.3. *The file my-experiment.js.*

```
1  /* recognition memory experiment set−up */
2  var my_experiment = function (){
3
4    instructions ('first delay phase (please press <esc> key to end): (please press any key to continue)')
5
6    delay_task ('please write out anything that comes to mind (please press <esc> key when finished) (
          please press any key to continue)')
7
8    /* instruction slide */
9    instructions ('second delay phase (5 seconds): (please press any key to continue)')
10
11   /* set up delay task: 5 seconds */
12   delay_task ('please type names of as many countries as you can think of in 5 seconds, separated by
          spaces.. (please press any key to continue)',
13             5000  /* 5000 mS */)
14
15   /* instruction slide */
16   /* instruction slide −− fixed duration */
17   var x = instructions ('thank you for completing the delay task: test phase coming up in 5 seconds.. ')
18   x.set_expiry (5000)
19   x.key_expiry = false
20
21   instructions ('third delay phase (10 seconds): (please press any key to continue)')
22
23   /* set up delay task: 10 seconds */
24   delay_task ('please type names of as many countries as you can think of in 6 seconds, separated by
          spaces.. (please press any key to continue)',
25             6000  /* 10000 mS */)
26
27   /* instruction slide */
28   instructions ('all done.. thank you.. (please press any key to continue)')
29 }
```

3.3. **experiments/feedback/my-experiment.js.** To collect user feedback in a multiple-choice format, we use the "feedback" command, which takes two parameters: 1) the text to be supplied to the user, and 2) an array of key-codes for the permitted responses.

- E.g.,
  - the array [49, 50, 51, 52, 53] represents the numeric keys 1-5
  - [49, 50, 51, 52, 53, 54, 55, 56, 57, 48] represents the numeric keys 0-9
  - [65, 66, 67, 68] represents the letters A,B,C,D

- and of course, other key combinations are possible. It may be necessary to consult a reference table such as
  - https://en.wikipedia.org/wiki/List_of_Unicode_characters#Basic_Latin

Please see the below file **my-experiment.js** for examples.

3.3.1. *The file my-experiment.js.*

```
1  /* recognition memory experiment set-up */
2  var my_experiment = function(){
3
4    /* instructions */
5    instructions('feedback coming up.. (please press any key to continue)')
6
7    /* feedback "task" */
8    feedback('please enter your affinity with the last stimulus on a scale of 1-5',
9          [49, 50, 51, 52, 53])
10
11   /* instructions */
12   instructions('thank you... more feedback coming up.. (please press any key to continue)')
13
14   /* more feedback "task" */
15   feedback('please enter your affinity with the last stimulus on a scale of 0-9',
16         [49, 50, 51, 52, 53, 54, 55, 56, 57, 48])
17
18   /* instructions */
19   instructions('thank you... multiple choice style feedback coming up.. (please press any key to
20       continue)')
21   /* feedback "task" */
22   feedback('skill testing question: 10*10 is: a) 100 b) 200 c) 1000 d) 10000',
23         [65, 66, 67, 68])
24
25   /* instructions */
26   instructions('thank you.. (please press any key to continue)')
27 }
```

3.4. **experiments/study-phase/my-experiment.js.** The study phase is the part of an experiment/survey where word/image (or other) stimuli are revealed in sequence. To implement a "study-phase", we must:

(1) declare a stimulus pool (e.g., line 9 of the file below)

(2) optionally, add a number of images to the stimulus pool (e.g., line 12 of the file below)

(3) optionally, add a number of words to the stimulus pool (e.g., lines 15-17 of the file below)

(4) make a selection of items from the stimulus pool (e.g., line 20 of the file below):

- note that only one parameter is supplied on line 20 (the parameter "N" from the spec), although it will be necessary for a "M" parameter to be included in subsequent examples, in order to implement the "test-phase". This will be discussed re: the next example.

(5) declare the study phase:

- note that declaring the study phase is as simple as (where "p" is a stimulus pool, previously defined in the **my-experiment.js** file):

  study_phase(p)

  although, in line 23 of the file below, two extra parameters are added:

  - an ISI (which was set to be 111 mS in the example): an (optional) duration of exposure of nothing, between exposure of stimuli, and

  - a SET (stimulus expiry time) which represents an (optional) maximum duration of exposure of the stimulus. Note: if one additional parameter is included in hte "study_phase" statement, this will be interpreted as ISI. In order to declare a "study_phase" with a SET (but not an ISI), one should provide two extra parameters (after "p") where the first parameter (the ISI) is set as 0, e.g.
    study_phase(p, 0, 5000)

    where the parameters above represent a "study_phase" with ISI of 0 mS, and a SET of 5 seconds.

3.4.1. *The file my-experiment.js.*

```
1  /* recognition memory experiment set-up */
2  var my_experiment = function(){
3
4    /* instructions */
5    instructions('study phase coming next: (please press any key to continue)')
6    instructions('please remember each word/image shown\n\n\neach word/image is displayed for up to 5
         seconds:\n\n\nif you are done with a particular word/image in less than 5 seconds, please press
         any key to advance to the next word/image\n\n\n(please press any key to continue)')
7
8    /* set up a stimulus pool */
9    var p = stimulus_pool()
10
11   /* add images to stimulus pool */
12   p.add_image(10)
13
14   /* add words to stimulus pool */
15   p.add('floccinaucinihilipilification')
16   p.add('supercalifragilisticexpialidocious')
17   p.add('umdiddlediddlediddleumdiddlei')
18
19   /* select portion of items from stimulus pool */
20   p.select(5)
21
22   /* set up 'study phase': show selected portions of pool */
23   study_phase(p,   /* stimulus pool */
24             111 /* ISI (optional) */,
25             5000  /* SET (optional) */ )
26 }
```

3.5. **experiments/test-phase/my-experiment.js.** For the "test-phase" example, we don't introduce anything new in lines 1-16.

However, on line 19, note that the second "M" parameter is added: calling the function **p.select()** with two parameters, as in line 19, is required before implementing the "study-phase" and "test-phase" in lines 22 and 28, respectively.

Note that, to declare a "test_phase" as on line 28 below, an optional ISI parameter has been included. The ISI parameter is optional, so the simplest invocation is:

  test_phase(p)

Similarly to declaring a "study-phase" in the previous example, a SET parameter may also be added when declaring a "test-phase" (again, if one wishes to declare a "test_phase" without ISI, and with SET > 0, two parameters need to be included: the first being ISI, which should be 0). So, while

test_phase(p)

is the simplest invocation,

test_phase(p, 111)

adds an ISI of 111 mS, and

test_phase(p, 111, 5000)

adds a SET of 5 seconds. Of course,

test_phase(p, 0, 5000)

would add a SET of 5 seconds, without adding ISI.

3.5.1. *The file my-experiment.js.*

```
1  /* recognition memory experiment set-up */
2  var my_experiment = function(){
3
4    /* set up some instruction slides */
5    instructions('study phase: please remember words/images and,\n\n\n\t* please press any key to advance
            to the next word/image\n\n\n(please press any key to continue)')
6
7    /* set up a stimulus pool */
8    var p = stimulus_pool()
9
10   /* add 10 available images to stimulus pool */
11   p.add_image(10)
12
13   /* add words to stimulus pool */
14   p.add('floccinaucinihilipilification')
15   p.add('supercalifragilisticexpialidocious')
16   p.add('umdiddlediddlediddleumdiddlei')
17
18   /* selection from stimulus pool: parameters are N, M as per the requirements */
19   p.select(5, 5)
20
21   /* set up 'study phase': show selected portions of pool */
22   study_phase(p, 111 /* ISI of 111 mS */ )
23
24   /* some instructions before 'test phase' */
25   instructions('test phase coming up:\n\n\nwhen you see an image/word, please press m or n:\n\n\n\t*
            please press m if you saw an image/word before\n\n\n\t* please press n if you did not see the
            image/word before\n\n\n(please press any key to continue)')
26
27   /* set up 'test phase' (user input recorded for whole randomized pool) */
28   test_phase(p, 333 /* ISI of 333 mS */ )
29 }
```

3.6. experiments/my-experiment/my-experiment.js. This example represents a more detailed sample, which is more representative of an "actual" experiment.

3.6.1. *Study/Test Phase: Multiple Stimuli Pools.* In this example, the procedure for adding stimulus to a pool is diversified by the specification of multiple stimulus pools, as in lines 8-16 for a first pool, and lines 19-27 for a second pool.

Note: the required selection from each of the two stimulus pools, happens at lines 30 and 31, respectively.

3.6.2. *The file my-experiment.js.*

```
1  /* recognition memory experiment set-up: customized/ complex experiment */
2  var my_experiment = function(){
3
4    /* set up some instruction slides */
5    instructions('study phase: please remember words/images and press any key (please press any key to
         continue)')
6
7    /* set up a stimulus pool */
8    var p1 = stimulus_pool()
9
10   /* add images to stimulus pool */
11   p1.add_image(10)
12
13   /* add words to stimulus pool */
14   p1.add('floccinaucinihilipilification')
15   p1.add('supercalifragilisticexpialidocious')
16   p1.add('equanimity')
17
18   /* set up a stimulus pool */
19   var p2 = stimulus_pool()
20
21   /* add images to stimulus pool */
22   p2.add_image(10)
23
24   /* add words to second stimulus pool */
25   p2.add('compassion')
26   p2.add('dogovarivatsya')
27   p2.add('umdiddlediddlediddleumdiddlei')
28
29   /* selection from stimulus pool (parameters are N, M) */
30   p1.select(5, 5)
31   p2.select(5, 5)
32
33   /* need to bundle the two pools together, into an array */
34   var two_pools = [p1, p2]
35
36   /* set up 'study phase': show selected portions of pool */
37   study_phase(two_pools,
38                111, /* ISI */
39                4500 /* SET */ )
40
41   /* instruction slide */
42   instructions('second delay phase (5 seconds): (please press any key to continue)')
43
44   /* set up delay task: 5 seconds */
45   delay_task('please type names of as many countries as you can think of in 10 seconds, separated by
         spaces.. (please press any key to continue)',
46                10000 /* 5000 mS */)
47
48   /* instruction slide -- fixed duration */
49   var x = instructions('thank you for completing the delay task: test phase coming up in 5 seconds..')
50   x.set_expiry(5000)
51   x.key_expiry = false
52
53   /* some instructions before 'test phase' */
54   instructions('test phase coming up (please press any key to continue)')
55   instructions('when you see an image/word, please press m or n (please press any key to continue)')
```

```
56    instructions('please press m if you saw an image/word before (please press any key to continue)')
57    instructions('please press n if you did not see the image/word before (please press any key to
          continue)')
58
59    /* set up 'test phase' (user input recorded for whole randomized pool) */
60    test_phase(two_pools, /* stimulus pools */
61              111, /* ISI */
62              6000, /* SET */
63              6, /* extra feedback (one for every 6 slides, approx.) */
64              "How did you feel about the last stimulus? A=positive, B=negative, C=neutral, D=not sure",
                  /* message for extra feedback */
65              [65, 66, 67, 68] /* accepted keypresses for extra feedback */ )
66 }
```

4. Sample Response Data

4.1. instructions.

```
1 url,event_id,task_id,task_type,trial_id,duration(mS),start(yyyy:mm:dd:hh:mn:ss:mls),end(yyyy:mm:dd:hh:
     mn:ss:mls),isi,set,stim_type,stim_id,stim_pool_id,response
2 http://ashy.ca/memory/experiments/instructions/memory.html,0,0,instructions
     ,0,7110.3,2017:50:20:22:37:50:980,2017:50:20:22:37:12:208,,,,,,""
3 http://ashy.ca/memory/experiments/instructions/memory.html,1,1,instructions
     ,0,1276.8,2017:50:20:22:37:12:208,2017:50:20:22:37:13:484,,,,,,""
4 http://ashy.ca/memory/experiments/instructions/memory.html,2,2,instructions
     ,0,590.6,2017:50:20:22:37:13:484,2017:50:20:22:37:14:750,,,,,,""
5 http://ashy.ca/memory/experiments/instructions/memory.html,3,3,instructions
     ,0,1052.9,2017:50:20:22:37:14:750,2017:50:20:22:37:15:128,,,,,,""
6 http://ashy.ca/memory/experiments/instructions/memory.html,4,4,instructions
     ,0,5003.9,2017:50:20:22:37:15:128,2017:50:20:22:37:20:132,,5000,,,,""
7 http://ashy.ca/memory/experiments/instructions/memory.html,5,5,instructions
     ,0,880.1,2017:50:20:22:37:20:132,2017:50:20:22:37:21:120,,5000,,,,""
8 http://ashy.ca/memory/experiments/instructions/memory.html,6,6,instructions
     ,0,676.8,2017:50:20:22:37:21:120,2017:50:20:22:37:21:689,,,,,,""
```

4.2. **delay.**

```
1 url , event_id , task_id , task_type , trial_id , duration (mS) , start (yyyy :mm: dd : hh :mn: ss : mls) , end (yyyy :mm: dd : hh :
    mn: ss : mls) , isi , set , stim_type , stim_id , stim_pool_id , response
2 http :// ashy . ca/memory/ experiments / delay /memory . html ,0 ,0 , instructions
    ,0 ,1496.8 ,2017:50:20:22:37:42:341 ,2017:50:20:22:37:43:838 , , , , , ,""
3 http :// ashy . ca/memory/ experiments / delay /memory . html ,1 ,1 , isi
    ,0 ,502.4 ,2017:50:20:22:37:43:838 ,2017:50:20:22:37:44:340 ,500 ,500 , , , ,""
4 http :// ashy . ca/memory/ experiments / delay /memory . html ,2 ,2 , instructions
    ,0 ,733.8 ,2017:50:20:22:37:44:340 ,2017:50:20:22:37:45:740 , , , , , ,""
5 http :// ashy . ca/memory/ experiments / delay /memory . html ,3 ,1 , delay
    ,0 ,7759.7 ,2017:50:20:22:37:45:740 ,2017:50:20:22:37:52:833 , , , , , ," typing something in here and
    pressing escape .."
6 http :// ashy . ca/memory/ experiments / delay /memory . html ,4 ,3 , instructions
    ,0 ,831.1 ,2017:50:20:22:37:52:834 ,2017:50:20:22:37:53:665 , , , , , ,""
7 http :// ashy . ca/memory/ experiments / delay /memory . html ,5 ,4 , isi
    ,0 ,505.5 ,2017:50:20:22:37:53:665 ,2017:50:20:22:37:54:170 ,500 ,500 , , , ,""
8 http :// ashy . ca/memory/ experiments / delay /memory . html ,6 ,5 , instructions
    ,0 ,1067.9 ,2017:50:20:22:37:54:170 ,2017:50:20:22:37:55:238 , , , , , ,""
9 http :// ashy . ca/memory/ experiments / delay /memory . html ,7 ,4 , delay
    ,0 ,5003.7 ,2017:50:20:22:37:55:238 ,2017:50:20:22:38:00:242 , ,5000 , , , ," peru india japan cyprus is"
10 http :// ashy . ca/memory/ experiments / delay /memory . html ,8 ,6 , instructions
    ,0 ,5003.2 ,2017:50:20:22:38:00:242 ,2017:50:20:22:38:50:245 , ,5000 , , , ,""
11 http :// ashy . ca/memory/ experiments / delay /memory . html ,9 ,7 , instructions
    ,0 ,1817.5 ,2017:50:20:22:38:50:245 ,2017:50:20:22:38:70:620 , , , , , ,""
12 http :// ashy . ca/memory/ experiments / delay /memory . html ,10 ,8 , isi
    ,0 ,503.9 ,2017:50:20:22:38:70:620 ,2017:50:20:22:38:70:566 ,500 ,500 , , , ,""
13 http :// ashy . ca/memory/ experiments / delay /memory . html ,11 ,9 , instructions
    ,0 ,2055.8 ,2017:50:20:22:38:70:566 ,2017:50:20:22:38:90:622 , , , , , ,""
14 http :// ashy . ca/memory/ experiments / delay /memory . html ,12 ,8 , delay
    ,0 ,6003.1 ,2017:50:20:22:38:90:622 ,2017:50:20:22:38:15:625 , ,6000 , , , ," canada bermuda panama germany "
15 http :// ashy . ca/memory/ experiments / delay /memory . html ,13 ,10 , instructions
    ,0 ,949.1 ,2017:50:20:22:38:15:625 ,2017:50:20:22:38:16:574 , , , , , ,""
```

4.3. study-phase.

```
1  url , event_id , task_id , task_type , trial_id , duration (mS) , start ( yyyy :mm: dd : hh :mn: ss : mls ) , end ( yyyy :mm: dd : hh :
       mn: ss : mls ) , isi , set , stim_type , stim_id , stim_pool_id , response
2  http :// ashy . ca/memory/ experiments / study−phase /memory . html , 0 , 0 , instructions
       , 0 , 845 , 2017:50:20:22:38:56:648 , 2017:50:20:22:38:57:493 , , , , , , ""
3  http :// ashy . ca/memory/ experiments / study−phase /memory . html , 1 , 1 , instructions
       , 0 , 629.5 , 2017:50:20:22:38:57:493 , 2017:50:20:22:38:58:122 , , , , , , ""
4  http :// ashy . ca/memory/ experiments / study−phase /memory . html , 2 , 2 , isi
       , 0 , 114.5 , 2017:50:20:22:38:58:123 , 2017:50:20:22:38:58:237 , 111 , 111 , , , 1 , ""
5  http :// ashy . ca/memory/ experiments / study−phase /memory . html , 3 , 2 , study_phase
       , 0 , 1369.5 , 2017:50:20:22:38:58:237 , 2017:50:20:22:38:59:607 , , 4500 , image , .. / .. / images /194. jpg , 1 , ""
6  http :// ashy . ca/memory/ experiments / study−phase /memory . html , 4 , 2 , isi
       , 1 , 111.9 , 2017:50:20:22:38:59:607 , 2017:50:20:22:38:59:719 , 111 , 111 , , , 1 , ""
7  http :// ashy . ca/memory/ experiments / study−phase /memory . html , 5 , 2 , study_phase
       , 1 , 654.6 , 2017:50:20:22:38:59:719 , 2017:50:20:22:39:00:373 , , 4500 , image , .. / .. / images /70. jpg , 1 , ""
8  http :// ashy . ca/memory/ experiments / study−phase /memory . html , 6 , 2 , isi
       , 2 , 113.3 , 2017:50:20:22:39:00:373 , 2017:50:20:22:39:00:486 , 111 , 111 , , , 1 , ""
9  http :// ashy . ca/memory/ experiments / study−phase /memory . html , 7 , 2 , study_phase
       , 2 , 547.7 , 2017:50:20:22:39:00:487 , 2017:50:20:22:39:10:340 , , 4500 , image , .. / .. / images /48. jpg , 1 , ""
10 http :// ashy . ca/memory/ experiments / study−phase /memory . html , 8 , 2 , isi
       , 3 , 115.2 , 2017:50:20:22:39:10:340 , 2017:50:20:22:39:10:149 , 111 , 111 , , , 1 , ""
11 http :// ashy . ca/memory/ experiments / study−phase /memory . html , 9 , 2 , study_phase
       , 3 , 521.8 , 2017:50:20:22:39:10:149 , 2017:50:20:22:39:10:671 , , 4500 , word , floccinaucinihilipilification
       , 1 , ""
12 http :// ashy . ca/memory/ experiments / study−phase /memory . html , 10 , 2 , isi
       , 4 , 113.2 , 2017:50:20:22:39:10:671 , 2017:50:20:22:39:10:785 , 111 , 111 , , , 1 , ""
13 http :// ashy . ca/memory/ experiments / study−phase /memory . html , 11 , 2 , study_phase
       , 4 , 598.8 , 2017:50:20:22:39:10:785 , 2017:50:20:22:39:20:383 , , 4500 , image , .. / .. / images /16. jpg , 1 , ""
```

4.4. **test-phase.**

```
 1  url, event_id, task_id, task_type, trial_id, duration(mS), start(yyyy:mm:dd:hh:mn:ss:mls), end(yyyy:mm:dd:hh:
        mn:ss:mls), isi, set, stim_type, stim_id, stim_pool_id, response
 2  http://ashy.ca/memory/experiments/test-phase/memory.html,0,0,instructions
        ,0,695.1,2017:50:20:22:39:19:512,2017:50:20:22:39:20:207,,,,,,""
 3  http://ashy.ca/memory/experiments/test-phase/memory.html,1,1,isi
        ,0,117.1,2017:50:20:22:39:20:208,2017:50:20:22:39:20:325,111,111,,,1,""
 4  http://ashy.ca/memory/experiments/test-phase/memory.html,2,1,study_phase
        ,0,356.3,2017:50:20:22:39:20:325,2017:50:20:22:39:20:681,,,image,../../images/16.jpg,1,""
 5  http://ashy.ca/memory/experiments/test-phase/memory.html,3,1,isi
        ,1,112.2,2017:50:20:22:39:20:681,2017:50:20:22:39:20:793,111,111,,,1,""
 6  http://ashy.ca/memory/experiments/test-phase/memory.html,4,1,study_phase
        ,1,344.3,2017:50:20:22:39:20:793,2017:50:20:22:39:21:138,,,word,floccinaucinihilipilification,1,""
 7  http://ashy.ca/memory/experiments/test-phase/memory.html,5,1,isi
        ,2,116.5,2017:50:20:22:39:21:138,2017:50:20:22:39:21:254,111,111,,,1,""
 8  http://ashy.ca/memory/experiments/test-phase/memory.html,6,1,study_phase
        ,2,319.3,2017:50:20:22:39:21:254,2017:50:20:22:39:21:573,,,image,../../images/48.jpg,1,""
 9  http://ashy.ca/memory/experiments/test-phase/memory.html,7,1,isi
        ,3,116.5,2017:50:20:22:39:21:573,2017:50:20:22:39:21:690,111,111,,,1,""
10  http://ashy.ca/memory/experiments/test-phase/memory.html,8,1,study_phase
        ,3,309.6,2017:50:20:22:39:21:690,2017:50:20:22:39:21:999,,,image,../../images/70.jpg,1,""
11  http://ashy.ca/memory/experiments/test-phase/memory.html,9,1,isi
        ,4,113.1,2017:50:20:22:39:22:000,2017:50:20:22:39:22:113,111,111,,,1,""
12  http://ashy.ca/memory/experiments/test-phase/memory.html,10,1,study_phase
        ,4,308,2017:50:20:22:39:22:113,2017:50:20:22:39:22:421,,,image,../../images/194.jpg,1,""
13  http://ashy.ca/memory/experiments/test-phase/memory.html,11,2,instructions
        ,0,410.7,2017:50:20:22:39:22:421,2017:50:20:22:39:22:831,,,,,,""
14  http://ashy.ca/memory/experiments/test-phase/memory.html,12,3,instructions
        ,0,616.1,2017:50:20:22:39:22:831,2017:50:20:22:39:23:447,,,,,,""
15  http://ashy.ca/memory/experiments/test-phase/memory.html,13,4,instructions
        ,0,316.1,2017:50:20:22:39:23:447,2017:50:20:22:39:23:763,,,,,,""
16  http://ashy.ca/memory/experiments/test-phase/memory.html,14,5,instructions
        ,0,321,2017:50:20:22:39:23:764,2017:50:20:22:39:24:840,,,,,,""
17  http://ashy.ca/memory/experiments/test-phase/memory.html,15,6,isi
        ,0,338.5,2017:50:20:22:39:24:840,2017:50:20:22:39:24:423,333,333,,,1,""
18  http://ashy.ca/memory/experiments/test-phase/memory.html,16,6,test_phase
        ,0,1054.7,2017:50:20:22:39:24:423,2017:50:20:22:39:25:478,,,image,../../images/48.jpg,1,"M"
19  http://ashy.ca/memory/experiments/test-phase/memory.html,17,6,isi
        ,1,338.4,2017:50:20:22:39:25:478,2017:50:20:22:39:25:816,333,333,,,1,""
20  http://ashy.ca/memory/experiments/test-phase/memory.html,18,6,test_phase
        ,1,1776.4,2017:50:20:22:39:25:816,2017:50:20:22:39:27:593,,,word,floccinaucinihilipilification,1,"M
        "
21  http://ashy.ca/memory/experiments/test-phase/memory.html,19,6,isi
        ,2,336.4,2017:50:20:22:39:27:593,2017:50:20:22:39:27:929,333,333,,,1,""
22  http://ashy.ca/memory/experiments/test-phase/memory.html,20,6,test_phase
        ,2,636.5,2017:50:20:22:39:27:929,2017:50:20:22:39:28:565,,,image,../../images/97.jpg,1,"N"
23  http://ashy.ca/memory/experiments/test-phase/memory.html,21,6,isi
        ,3,338.4,2017:50:20:22:39:28:565,2017:50:20:22:39:28:904,333,333,,,1,""
24  http://ashy.ca/memory/experiments/test-phase/memory.html,22,6,test_phase
        ,3,714.4,2017:50:20:22:39:28:904,2017:50:20:22:39:29:618,,,image,../../images/70.jpg,1,"M"
25  http://ashy.ca/memory/experiments/test-phase/memory.html,23,6,isi
        ,4,337,2017:50:20:22:39:29:618,2017:50:20:22:39:29:955,333,333,,,1,""
26  http://ashy.ca/memory/experiments/test-phase/memory.html,24,6,test_phase
        ,4,659.9,2017:50:20:22:39:29:955,2017:50:20:22:39:30:615,,,image,../../images/29.jpg,1,"N"
27  http://ashy.ca/memory/experiments/test-phase/memory.html,25,6,isi
        ,5,336.6,2017:50:20:22:39:30:615,2017:50:20:22:39:30:952,333,333,,,1,""
28  http://ashy.ca/memory/experiments/test-phase/memory.html,26,6,test_phase
        ,5,625.7,2017:50:20:22:39:30:952,2017:50:20:22:39:31:577,,,image,../../images/42.jpg,1,"N"
29  http://ashy.ca/memory/experiments/test-phase/memory.html,27,6,isi
        ,6,338.4,2017:50:20:22:39:31:577,2017:50:20:22:39:31:916,333,333,,,1,""
30  http://ashy.ca/memory/experiments/test-phase/memory.html,28,6,test_phase
        ,6,1009.4,2017:50:20:22:39:31:916,2017:50:20:22:39:32:925,,,word,supercalifragilisticexpialidocious
        ,1,"N"
31  http://ashy.ca/memory/experiments/test-phase/memory.html,29,6,isi
        ,7,334.3,2017:50:20:22:39:32:925,2017:50:20:22:39:33:259,333,333,,,1,""
32  http://ashy.ca/memory/experiments/test-phase/memory.html,30,6,test_phase
        ,7,577.7,2017:50:20:22:39:33:260,2017:50:20:22:39:33:837,,,image,../../images/194.jpg,1,"M"
```

33 http://ashy.ca/memory/experiments/test-phase/memory.html,31,6,isi
        ,8,338.5,2017:50:20:22:39:33:837,2017:50:20:22:39:34:176,333,333,,,1,""
34 http://ashy.ca/memory/experiments/test-phase/memory.html,32,6,test_phase
        ,8,634.4,2017:50:20:22:39:34:176,2017:50:20:22:39:34:810,,,image,../../images/16.jpg,1,"M"
35 http://ashy.ca/memory/experiments/test-phase/memory.html,33,6,isi
        ,9,335.3,2017:50:20:22:39:34:810,2017:50:20:22:39:35:145,333,333,,,1,""
36 http://ashy.ca/memory/experiments/test-phase/memory.html,34,6,test_phase
        ,9,606.8,2017:50:20:22:39:35:145,2017:50:20:22:39:35:752,,,image,../../images/34.jpg,1,"N"
37 http://ashy.ca/memory/experiments/test-phase/memory.html,35,7,instructions
        ,0,861.8,2017:50:20:22:39:35:752,2017:50:20:22:39:36:614,,,,,,""

4.5. **my-experiment.**

```
 1  url , event_id , task_id , task_type , trial_id , duration (mS) , start (yyyy :mm: dd : hh :mn: ss : mls ) , end (yyyy :mm: dd : hh :
        mn: ss : mls ) , isi , set , stim_type , stim_id , stim_pool_id , response
 2  http :// ashy . ca/memory/ experiments /my−experiment /memory . html ,0 ,0 , instructions
        ,0 ,1630.4 ,2017:50:20:22:39:56:566 ,2017:50:20:22:39:58:196 , , , , , ,""
 3  http :// ashy . ca/memory/ experiments /my−experiment /memory . html ,1 ,1 , isi
        ,0 ,112.2 ,2017:50:20:22:39:58:196 ,2017:50:20:22:39:58:308 ,111 ,111 , , ,2 ,""
 4  http :// ashy . ca/memory/ experiments /my−experiment /memory . html ,2 ,1 , study_phase
        ,0 ,763.9 ,2017:50:20:22:39:58:308 ,2017:50:20:22:39:59:720 , ,4500 , image , . . / . . / images /198. jpg ,2 ,""
 5  http :// ashy . ca/memory/ experiments /my−experiment /memory . html ,3 ,1 , isi
        ,1 ,112.3 ,2017:50:20:22:39:59:720 ,2017:50:20:22:39:59:185 ,111 ,111 , , ,2 ,""
 6  http :// ashy . ca/memory/ experiments /my−experiment /memory . html ,4 ,1 , study_phase
        ,1 ,689.2 ,2017:50:20:22:39:59:185 ,2017:50:20:22:39:59:874 , ,4500 , image , . . / . . / images /186. jpg ,2 ,""
 7  http :// ashy . ca/memory/ experiments /my−experiment /memory . html ,5 ,1 , isi
        ,2 ,116.5 ,2017:50:20:22:39:59:874 ,2017:50:20:22:39:59:990 ,111 ,111 , , ,2 ,""
 8  http :// ashy . ca/memory/ experiments /my−experiment /memory . html ,6 ,1 , study_phase
        ,2 ,340.3 ,2017:50:20:22:39:59:991 ,2017:50:20:22:40:00:331 , ,4500 , image , . . / . . / images /48. jpg ,1 ,""
 9  http :// ashy . ca/memory/ experiments /my−experiment /memory . html ,7 ,1 , isi
        ,3 ,114.2 ,2017:50:20:22:40:00:331 ,2017:50:20:22:40:00:445 ,111 ,111 , , ,2 ,""
10  http :// ashy . ca/memory/ experiments /my−experiment /memory . html ,8 ,1 , study_phase
        ,3 ,285.7 ,2017:50:20:22:40:00:445 ,2017:50:20:22:40:00:731 , ,4500 , image , . . / . . / images /16. jpg ,1 ,""
11  http :// ashy . ca/memory/ experiments /my−experiment /memory . html ,9 ,1 , isi
        ,4 ,114.7 ,2017:50:20:22:40:00:731 ,2017:50:20:22:40:00:845 ,111 ,111 , , ,2 ,""
12  http :// ashy . ca/memory/ experiments /my−experiment /memory . html ,10 ,1 , study_phase
        ,4 ,306.6 ,2017:50:20:22:40:00:846 ,2017:50:20:22:40:10:152 , ,4500 , image , . . / . . / images /73. jpg ,2 ,""
13  http :// ashy . ca/memory/ experiments /my−experiment /memory . html ,11 ,1 , isi
        ,5 ,113.4 ,2017:50:20:22:40:10:152 ,2017:50:20:22:40:10:265 ,111 ,111 , , ,2 ,""
14  http :// ashy . ca/memory/ experiments /my−experiment /memory . html ,12 ,1 , study_phase
        ,5 ,257.7 ,2017:50:20:22:40:10:265 ,2017:50:20:22:40:10:523 , ,4500 , image , . . / . . / images /194. jpg ,1 ,""
15  http :// ashy . ca/memory/ experiments /my−experiment /memory . html ,13 ,1 , isi
        ,6 ,116.5 ,2017:50:20:22:40:10:523 ,2017:50:20:22:40:10:640 ,111 ,111 , , ,2 ,""
16  http :// ashy . ca/memory/ experiments /my−experiment /memory . html ,14 ,1 , study_phase
        ,6 ,304.2 ,2017:50:20:22:40:10:640 ,2017:50:20:22:40:10:944 , ,4500 , word , dogovarivatsya ,2 ,""
17  http :// ashy . ca/memory/ experiments /my−experiment /memory . html ,15 ,1 , isi
        ,7 ,113.8 ,2017:50:20:22:40:10:944 ,2017:50:20:22:40:20:580 ,111 ,111 , , ,2 ,""
18  http :// ashy . ca/memory/ experiments /my−experiment /memory . html ,16 ,1 , study_phase
        ,7 ,297.1 ,2017:50:20:22:40:20:580 ,2017:50:20:22:40:20:355 , ,4500 , image , . . / . . / images /70. jpg ,1 ,""
19  http :// ashy . ca/memory/ experiments /my−experiment /memory . html ,17 ,1 , isi
        ,8 ,112.4 ,2017:50:20:22:40:20:355 ,2017:50:20:22:40:20:467 ,111 ,111 , , ,2 ,""
20  http :// ashy . ca/memory/ experiments /my−experiment /memory . html ,18 ,1 , study_phase
        ,8 ,278.6 ,2017:50:20:22:40:20:467 ,2017:50:20:22:40:20:746 , ,4500 , image , . . / . . / images /170. jpg ,2 ,""
21  http :// ashy . ca/memory/ experiments /my−experiment /memory . html ,19 ,1 , isi
        ,9 ,114.6 ,2017:50:20:22:40:20:746 ,2017:50:20:22:40:20:860 ,111 ,111 , , ,2 ,""
22  http :// ashy . ca/memory/ experiments /my−experiment /memory . html ,20 ,1 , study_phase
        ,9 ,281.2 ,2017:50:20:22:40:20:861 ,2017:50:20:22:40:30:142 , ,4500 , word , floccinaucinihilipilification
        ,1 ,""
23  http :// ashy . ca/memory/ experiments /my−experiment /memory . html ,21 ,2 , instructions
        ,0 ,411 ,2017:50:20:22:40:30:142 ,2017:50:20:22:40:30:553 , , , , , ,""
24  http :// ashy . ca/memory/ experiments /my−experiment /memory . html ,22 ,3 , isi
        ,0 ,503.8 ,2017:50:20:22:40:30:553 ,2017:50:20:22:40:40:570 ,500 ,500 , , , ,""
25  http :// ashy . ca/memory/ experiments /my−experiment /memory . html ,23 ,4 , instructions
        ,0 ,423.2 ,2017:50:20:22:40:40:570 ,2017:50:20:22:40:40:480 , , , , , ,""
26  http :// ashy . ca/memory/ experiments /my−experiment /memory . html ,24 ,3 , delay
        ,0 ,10001.2 ,2017:50:20:22:40:40:480 ,2017:50:20:22:40:14:481 , ,10000 , , , ,"canada chile argentina
        antarctica "
27  http :// ashy . ca/memory/ experiments /my−experiment /memory . html ,25 ,5 , instructions
        ,0 ,5001.5 ,2017:50:20:22:40:14:481 ,2017:50:20:22:40:19:482 , ,5000 , , , ,""
28  http :// ashy . ca/memory/ experiments /my−experiment /memory . html ,26 ,6 , instructions
        ,0 ,12886.2 ,2017:50:20:22:40:19:482 ,2017:50:20:22:40:32:368 , , , , , ,""
29  http :// ashy . ca/memory/ experiments /my−experiment /memory . html ,27 ,7 , instructions
        ,0 ,544.8 ,2017:50:20:22:40:32:368 ,2017:50:20:22:40:32:913 , , , , , ,""
30  http :// ashy . ca/memory/ experiments /my−experiment /memory . html ,28 ,8 , instructions
        ,0 ,396.2 ,2017:50:20:22:40:32:913 ,2017:50:20:22:40:33:309 , , , , , ,""
31  http :// ashy . ca/memory/ experiments /my−experiment /memory . html ,29 ,9 , instructions
        ,0 ,320.5 ,2017:50:20:22:40:33:309 ,2017:50:20:22:40:33:629 , , , , , ,""
32  http :// ashy . ca/memory/ experiments /my−experiment /memory . html ,30 ,10 , isi
        ,0 ,116.5 ,2017:50:20:22:40:33:629 ,2017:50:20:22:40:33:746 ,111 ,111 , , ,2 ,""
```

33 http://ashy.ca/memory/experiments/my−experiment/memory.html,31,10,test_phase
,0,911.2,2017:50:20:22:40:33:746,2017:50:20:22:40:34:657,,6000,image,../../images/78.jpg,2,"N"
34 http://ashy.ca/memory/experiments/my−experiment/memory.html,32,10,isi
,1,115.3,2017:50:20:22:40:34:657,2017:50:20:22:40:34:773,111,111,,,2,""
35 http://ashy.ca/memory/experiments/my−experiment/memory.html,33,10,test_phase
,1,1388.4,2017:50:20:22:40:34:773,2017:50:20:22:40:36:161,,6000,image,../../images/186.jpg,2,"M"
36 http://ashy.ca/memory/experiments/my−experiment/memory.html,34,10,isi
,2,114.2,2017:50:20:22:40:36:161,2017:50:20:22:40:36:275,111,111,,,1,""
37 http://ashy.ca/memory/experiments/my−experiment/memory.html,35,10,test_phase
,2,632.6,2017:50:20:22:40:36:275,2017:50:20:22:40:36:908,,6000,image,../../images/16.jpg,1,"M"
38 http://ashy.ca/memory/experiments/my−experiment/memory.html,36,10,isi
,3,113.3,2017:50:20:22:40:36:908,2017:50:20:22:40:37:210,111,111,,,2,""
39 http://ashy.ca/memory/experiments/my−experiment/memory.html,37,10,test_phase
,3,673.8,2017:50:20:22:40:37:210,2017:50:20:22:40:37:695,,6000,word,dogovarivatsya,2,"M"
40 http://ashy.ca/memory/experiments/my−experiment/memory.html,38,10,isi
,4,115.6,2017:50:20:22:40:37:695,2017:50:20:22:40:37:810,111,111,,,1,""
41 http://ashy.ca/memory/experiments/my−experiment/memory.html,39,10,test_phase
,4,560.5,2017:50:20:22:40:37:810,2017:50:20:22:40:38:371,,6000,image,../../images/29.jpg,1,"N"
42 http://ashy.ca/memory/experiments/my−experiment/memory.html,40,10,isi
,5,113.2,2017:50:20:22:40:38:371,2017:50:20:22:40:38:484,111,111,,,1,""
43 http://ashy.ca/memory/experiments/my−experiment/memory.html,41,10,test_phase
,5,633.7,2017:50:20:22:40:38:484,2017:50:20:22:40:39:118,,6000,image,../../images/34.jpg,1,"N"
44 http://ashy.ca/memory/experiments/my−experiment/memory.html,42,10,isi
,6,115.4,2017:50:20:22:40:39:118,2017:50:20:22:40:39:233,111,111,,,2,""
45 http://ashy.ca/memory/experiments/my−experiment/memory.html,43,10,test_phase
,6,1032.6,2017:50:20:22:40:39:233,2017:50:20:22:40:40:266,,6000,image,../../images/73.jpg,2,"N"
46 http://ashy.ca/memory/experiments/my−experiment/memory.html,44,10,isi
,7,114.7,2017:50:20:22:40:40:266,2017:50:20:22:40:40:381,111,111,,,1,""
47 http://ashy.ca/memory/experiments/my−experiment/memory.html,45,10,test_phase
,7,1123.3,2017:50:20:22:40:40:381,2017:50:20:22:40:41:504,,6000,word,floccinaucinihilipilification
,1,"M"
48 http://ashy.ca/memory/experiments/my−experiment/memory.html,46,10,isi
,8,114.3,2017:50:20:22:40:41:504,2017:50:20:22:40:41:618,111,111,,,1,""
49 http://ashy.ca/memory/experiments/my−experiment/memory.html,47,10,test_phase
,8,702.7,2017:50:20:22:40:41:618,2017:50:20:22:40:42:321,,6000,image,../../images/48.jpg,1,"M"
50 http://ashy.ca/memory/experiments/my−experiment/memory.html,48,10,isi
,9,111.9,2017:50:20:22:40:42:321,2017:50:20:22:40:42:433,111,111,,,2,""
51 http://ashy.ca/memory/experiments/my−experiment/memory.html,49,10,test_phase
,9,570,2017:50:20:22:40:42:433,2017:50:20:22:40:43:300,,6000,image,../../images/9.jpg,2,"N"
52 http://ashy.ca/memory/experiments/my−experiment/memory.html,50,11,feedback
,0,1422.9,2017:50:20:22:40:43:300,2017:50:20:22:40:44:426,,,,,,"A"
53 http://ashy.ca/memory/experiments/my−experiment/memory.html,51,10,isi
,10,114.3,2017:50:20:22:40:44:426,2017:50:20:22:40:44:540,111,111,,,2,""
54 http://ashy.ca/memory/experiments/my−experiment/memory.html,52,10,test_phase
,10,1323.7,2017:50:20:22:40:44:540,2017:50:20:22:40:45:864,,6000,image,../../images/80.jpg,2,"N"
55 http://ashy.ca/memory/experiments/my−experiment/memory.html,53,10,isi
,11,117.2,2017:50:20:22:40:45:864,2017:50:20:22:40:45:981,111,111,,,1,""
56 http://ashy.ca/memory/experiments/my−experiment/memory.html,54,10,test_phase
,11,855.8,2017:50:20:22:40:45:981,2017:50:20:22:40:46:836,,6000,word,
supercalifragilisticexpialidocious,1,"N"
57 http://ashy.ca/memory/experiments/my−experiment/memory.html,55,10,isi
,12,114.5,2017:50:20:22:40:46:836,2017:50:20:22:40:46:951,111,111,,,1,""
58 http://ashy.ca/memory/experiments/my−experiment/memory.html,56,10,test_phase
,12,601.6,2017:50:20:22:40:46:951,2017:50:20:22:40:47:553,,6000,image,../../images/97.jpg,1,"N"
59 http://ashy.ca/memory/experiments/my−experiment/memory.html,57,10,isi
,13,115.5,2017:50:20:22:40:47:553,2017:50:20:22:40:47:668,111,111,,,1,""
60 http://ashy.ca/memory/experiments/my−experiment/memory.html,58,10,test_phase
,13,897.5,2017:50:20:22:40:47:668,2017:50:20:22:40:48:566,,6000,image,../../images/194.jpg,1,"M"
61 http://ashy.ca/memory/experiments/my−experiment/memory.html,59,12,feedback
,0,2531,2017:50:20:22:40:48:566,2017:50:20:22:40:51:970,,,,,,"C"
62 http://ashy.ca/memory/experiments/my−experiment/memory.html,60,10,isi
,14,114.8,2017:50:20:22:40:51:970,2017:50:20:22:40:51:211,111,111,,,2,""
63 http://ashy.ca/memory/experiments/my−experiment/memory.html,61,10,test_phase
,14,1353,2017:50:20:22:40:51:211,2017:50:20:22:40:52:564,,6000,word,compassion,2,"N"
64 http://ashy.ca/memory/experiments/my−experiment/memory.html,62,13,feedback
,0,1654.9,2017:50:20:22:40:52:564,2017:50:20:22:40:54:219,,,,,,"C"
65 http://ashy.ca/memory/experiments/my−experiment/memory.html,63,10,isi
,15,112.9,2017:50:20:22:40:54:219,2017:50:20:22:40:54:332,111,111,,,1,""

66 http://ashy.ca/memory/experiments/my−experiment/memory.html,64,10,test_phase
,15,794,2017:50:20:22:40:54:332,2017:50:20:22:40:55:126,,6000,image,../../images/70.jpg,1,"M"
67 http://ashy.ca/memory/experiments/my−experiment/memory.html,65,14,feedback
,0,972.1,2017:50:20:22:40:55:126,2017:50:20:22:40:56:980,,,,,,"A"
68 http://ashy.ca/memory/experiments/my−experiment/memory.html,66,10,isi
,16,111.9,2017:50:20:22:40:56:980,2017:50:20:22:40:56:210,111,111,,,2,""
69 http://ashy.ca/memory/experiments/my−experiment/memory.html,67,10,test_phase
,16,1241.4,2017:50:20:22:40:56:210,2017:50:20:22:40:57:451,,6000,image,../../images/170.jpg,2,"M"
70 http://ashy.ca/memory/experiments/my−experiment/memory.html,68,10,isi
,17,115.7,2017:50:20:22:40:57:451,2017:50:20:22:40:57:567,111,111,,,1,""
71 http://ashy.ca/memory/experiments/my−experiment/memory.html,69,10,test_phase
,17,661.3,2017:50:20:22:40:57:567,2017:50:20:22:40:58:228,,6000,image,../../images/42.jpg,1,"N"
72 http://ashy.ca/memory/experiments/my−experiment/memory.html,70,10,isi
,18,115.1,2017:50:20:22:40:58:228,2017:50:20:22:40:58:343,111,111,,,2,""
73 http://ashy.ca/memory/experiments/my−experiment/memory.html,71,10,test_phase
,18,871.7,2017:50:20:22:40:58:343,2017:50:20:22:40:59:215,,6000,word,umdiddlediddlediddleumdiddlei
,2,"N"
74 http://ashy.ca/memory/experiments/my−experiment/memory.html,72,10,isi
,19,111.8,2017:50:20:22:40:59:215,2017:50:20:22:40:59:327,111,111,,,2,""
75 http://ashy.ca/memory/experiments/my−experiment/memory.html,73,10,test_phase
,19,751.1,2017:50:20:22:40:59:327,2017:50:20:22:41:00:780,,6000,image,../../images/198.jpg,2,"M"
76 http://ashy.ca/memory/experiments/my−experiment/memory.html,74,15,instructions
,0,1147,2017:50:20:22:41:00:780,2017:50:20:22:41:10:225,,,,,,""

5. Procedures

Add a new experiment

Assume images are numbered

ISI: omit or set to zero

Note: response text recorded is final output (not data stream).

Filtered for newline characters

6. Source Code: Client Side

6.1. egg-timer.js.

```
1  /* via developer.mozilla.org/en-US/docs/Web/API/WindowOrWorkerGlobalScope/clearTimeout */
2  var egg_timer = {
3
4    /* callback */
5    setup: function(t_ms){
6
7      /* assert parameter is a number */
8      if(typeof this.timeoutID === "number"){
9        this.cancel()
10     }
11
12     /* what to do when the timer expires */
13     this.timeoutID = window.setTimeout(
14       function(){
15         var now = ctx.get_state()
16         var id = now.id
17         now.ding = true
18         if(now.key_expiry == false || now.expiry_ms > 0){
19           now.expire()
20         }
21       }.bind(this), t_ms
22     )
23   }, cancel: function(){
24     window.clearTimeout(this.timeoutID)
25     this.timeoutID = undefined
26   }
27 }
```

6.2. **key.js.**

```
1  var bell = new Audio("../../ding.mp3")
2
3  /* convert from unicode to familiar symbol */
4  function unicode_from_key_event(e){
5    return e.charCode ? e.charCode : e.keyCode
6  }
7
8  /* keyboard status array (unicode format) */
9  var key_unicode = {}
10
11 /* keyboard event handler function */
12 function keyboard_module(){
13
14   /* set up key-down event handler function */
15   document.onkeydown = function(e){
16
17     /* unicode vs. character representation */
18     var unicode = unicode_from_key_event(e), key = String.fromCharCode(unicode)
19
20     /* inverted question mark */
21     if(unicode == 191){
22       unicode = 63, key = '?'
23     }else if(unicode == 188){
24       unicode = 44, key = ','
25     }else if(unicode == 190){
26       unicode = 46, key = "."
27     }else if(unicode == 13){
28
29       /* replace enter with space */
30       unicode = 32, key = " "
31     }
32
33     if(unicode == 27){
34
35       /* do nothing if we get a key that is code 27, but not an escape key.. */
36       if(!(e.key == "Escape" || e.key == "Esc")){
37         return;
38       }
39     }
40
41     if(unicode == 222){
42       unicode = 39, key ="'"
43     }
44
45     /* console.log("unicode", unicode) */
46
47     key_unicode[unicode] = true
48
49     var ignore = [20, 192, 189, 187, 93, 91, 219, 221, 222, 220, 186, 33, 36, 34, 35, 37, 38, 40]
50
51     /* ignore caps-lock and other special key */
52     if(ignore.includes(unicode)){
53       return
54     }
55
56     var allow = [];
57     for(var i=65; i<=90; i++){
58       allow.push(i);
59     }
60     for(var i=48; i<=57; i++){
61       allow.push(i);
62     }
63
64     /* allow space bar */
65     allow.push(32)
66
```

```
67      /* allow escape key */
68      allow.push(27)
69
70      /* allow comma */
71      allow.push(44)
72
73      /* allow period */
74      allow.push(46)
75
76      /* allow question mark */
77      allow.push(63)
78
79      /* allow backspace */
80      allow.push(8)
81
82      /* allow single right quotation mark */
83      allow.push(39)
84
85      if(!allow.includes(unicode)){
86        return
87      }
88
89      /* when are we? */
90      var now = ctx.get_state()
91
92      /* record key press, if admissible */
93      var admissible_keys = now.get_admissible_keys()
94      if(admissible_keys.includes(unicode) || now.type == 'delay'){
95        now.record_key_stroke(unicode)
96      }
97
98      /* by default, transition from a slide upon key-press */
99      var go = true
100
101     /* special treatment for delay task */
102     if(now.type == 'delay'){
103       if(now.txt == null){
104
105         /* init */
106         now.txt = ''
107       }
108       if(unicode == 8){
109
110         /* backspace */
111         var len = now.txt.length
112         now.txt = now.txt.substring(0, len - 1)
113
114       }else if(admissible_keys.includes(27) && unicode == 27){
115
116         /* break out of free-form text input mode with <esc> key */
117         ctx.clear_tmr()
118         now.expire()
119         // bell.play()
120
121         return key_unicode
122
123       }else{
124
125         /* add character to buffer */
126         if(unicode >= 65 && unicode <= 90){
127           now.txt += key.toLowerCase()
128         }else{
129           now.txt += key
130         }
131       }
132
133       /* redraw */
134       update()
```

```
135      }
136
137      /* check if this state "requires" keyboard input */
138      if(now.require_key() == true){
139
140        /* is the key that was pressed, in the list of "admissible" keys? */
141        if(admissible_keys.includes(unicode)){
142
143          /* if we have a "deja-vu" variable, calculate a score */
144          if(!(now.deja == undefined)){
145            ctx.questions_total += 1
146
147            /* check for N or M keypress */
148            if((now.deja == true && unicode == 77) || (now.deja == false && unicode == 78)){
149              ctx.questions_correct += 1
150            }
151          }
152        }else{
153          /* block if a key was required but the one entered was not admissible */
154          go = false
155        }
156      }
157
158      /* t <-- t + 1 */
159      if(now && now.key_expiry && go){
160
161        /* clear the timer and "go next" */
162        ctx.clear_tmr()
163        now.expire()
164      }
165    }
166    return key_unicode
167 }
```

6.3. **main.js.**

```
1  var abs_path = '../../', ctx = canvas.getContext("2d")
2
3  /* background color, shape parameter and font size */
4  document.bgColor = "#FFFFFF", ctx.pad = 20, ctx.font_size = 30
5
6  /* canvas dimensions manipulation */
7  var less = function(x){
8    return x - ctx.pad
9  }
10
11 ctx.w = function(){
12   return less(window.innerWidth)
13 }
14
15 ctx.h = function(){
16   return less(window.innerHeight)
17 }
18
19 /* canvas resize */
20 function resize(){
21   canvas.width = ctx.w(), canvas.height = ctx.h()
22 }
23
24 /* load corporate logo */
25 ctx.symbol = new Image()
26 ctx.symbol.fn = abs_path + "logo/uvic_gray.png"
27
28 /* algo to draw scaled corporate logo */
29 ctx.draw_symbol = function(){
30   var s_f = 5, pad = this.pad, s = this.symbol
31   var ww = window.innerWidth, wh = window.innerHeight
32   var w = ww - pad, h = wh - pad, w_s = s.width, h_s = s.height
33   var wf = (ww - pad) / (s_f * w_s), lwf = w_s * wf, lhf = h_s * wf
34   this.drawImage(s, w - lwf, h - lhf, lwf, lhf)
35 }
36
37 /* access current "state" (a state represents a particular "trial" in an experiment) */
38 ctx.set_state = function(s){
39   last_state = null
40   if(ctx.current_state != null){
41     last_state = ctx.current_state
42   }
43   ctx.current_state = s
44
45   /* sanity check */
46   if(s != null){
47     s.daddy = last_state
48   }
49   return(s)
50 }
51
52 /* access present "state" */
53 ctx.get_state = function(){
54   return ctx.current_state
55 }
56
57 /* trigger update/plotting from window resize event */
58 window.onresize = function(event){
59   update()
60 }
61
62 /* update the canvas (present the current "trial") */
63 function update(){
64   resize()
65   var now = ctx.get_state()
66   if(now){
```

```
67       now.show(ctx)
68    }
69 }
70
71 /* "in" hook: plot the current trial */
72 window.onload = function(){
73    update()
74 }
75
76 /* set up timer to coordinate transitions between trials */
77 ctx.egg_timer = egg_timer
78
79 ctx.clear_tmr = function(){
80    ctx.egg_timer.cancel()
81 }
82
83 ctx.init_tmr = function(t_ms){
84    ctx.egg_timer.setup(t_ms)
85 }
86
87 /* initialize reference to first and most-recently-initialized trials */
88 ctx.last_new_state = null, ctx.first_new_state = null
89
90 /* count number of questions answered correctly (this is redundant) */
91 ctx.questions_correct = 0, ctx.questions_total = 0
92
93 /* this function sets up the experiment (according to the user function my_experiment)
94 and we trigger this function after all the images have loaded. */
95 function run_before_loading_images(){
96
97    /* set up an experiment according to user specs/code */
98    my_experiment(ctx)
99
100   /* display a goodbye message every time */
101   instructions('survey complete: thank you for your participation')
102
103   ctx.last_state = ctx.last_new_state, ctx.first_state = ctx.first_new_state
104
105   /* start at the very beginning, it's a very good place to start.. */
106   ctx.set_state(ctx.first_state)
107
108   /* respond to keyboard events */
109   key_unicode = keyboard_module()
110
111   /* start "stopwatch" */
112   ctx.t0 = window.performance.now()
113
114 }
115
116
117 /* load some image files: need to change if the image database changes */
118 var n_imgs = 200, n_imgs_to_load = 0, n_imgs_loaded = 0
119
120 var images_to_load = []
121
122 /* scan images to determine which need to be loaded */
123 var idx = new Array()
124 ctx.imgs = new Array()
125 for(var i = 1; i <= n_imgs; i++){
126     idx.push(i)
127 }
128
129 /* randomize the order of the images */
130 shuffle(idx)
131
132 for(var i=1; i<=n_imgs; i++){
133   var img = new Image()
134   img.fn = abs_path + 'images/' + idx[i-1] + '.jpg'    // load_img(img) //var my_img = load_img(img_fn)
```

```
135    ctx.imgs.push(img)
136  }
137
138  var get_image = function(){
139      return ctx.imgs[n_imgs_to_load++]
140  }
141
142  /* load image data */
143  function load_img(i){
144      ctx.imgs[i].onload = function(){
145
146          /* have all images been loaded? */
147          if(++n_imgs_loaded == n_imgs_to_load){
148
149              /* proceed to init the experiment */
150              ctx.get_state().start()
151          }
152      }
153
154      /* load the image */
155      ctx.imgs[i].src = ctx.imgs[i].fn
156      return ctx.imgs[i]
157  }
158
159
160  /* keep track of the "task-index" as the experiment is intialized */
161  var next_task_id = 0
162
163  run_before_loading_images()
164
165
166  /* load the symbol */
167  ++ n_imgs_to_load
168
169  ctx.symbol.onload = function(){
170
171      /* have all images been loaded? */
172      if(++n_imgs_loaded == n_imgs_to_load){
173
174          /* proceed to init the experiment */
175          ctx.get_state().start()
176      }
177  }
178  ctx.symbol.src = ctx.symbol.fn
179
180  /* load the other images.. */
181  for(var i=0; i<ctx.imgs.length; i++){
182      if(ctx.imgs[i].load_me){
183          load_img(i)
184      }
185  }
```

6.4. **memory.js.**

```
 1  /* sleep function */
 2  function sleep(ms){
 3    return new Promise(resolve => setTimeout(resolve, ms))
 4  }
 5
 6  var js_added = -1, deps = []
 7
 8  /* j4v4scr1pt 4n4l0g 0f 1nclud3 st4t3m3nt */
 9  function add_js(fn){
10    var body = document.getElementsByTagName('body')[0], s = document.createElement('script')
11    s.async = false, s.src = fn + '.js'
12
13    /* wait until script is loaded before proceeding.. */
14    s.onload = function(){
15      if(++js_added < deps.length){
16        add_js(deps[js_added])
17      }
18    }
19    body.appendChild(s)
20  }
21
22  /* c411 411 th3 ch1ldr3n */
23  dependencies = ['text', 'key', 'util', 'task', 'pool', 'state', 'egg-timer']
24  for(var d in dependencies){
25    deps.push('../../' + dependencies[d])
26  }
27  deps.push('my-experiment')
28  deps.push('../../main')
29  add_js(deps[0], '')
```

6.5. **pool.js.**

```
1  var next_pool_id = 0
2
3  /* stimulus pool - object that has words or images added to it. Selections drawn randomly for "study
       phase"
4  by draw() method. That selection is shuffled back into the deck, for the "test phase" */
5  function pool(){
6
7    /* keep count */
8    ++ next_pool_id
9
10   this.is_pool = true, this.pool_id = next_pool_id, this.ctx = ctx, this.stimuli = new Array()
11
12   /* add a stimulus to the pool */
13   this.add = function(stim){
14     this.stimuli.push(stim)
15     stim.load_me = true
16     return stim
17   }
18
19   /* add one or more images to the stimulus pool */
20   this.add_image = function(n=1){
21     for(var i = 0; i < n; i++){
22       this.add(get_image())
23     }
24   }
25
26   /* set number of samples for study phase */
27   this.set_n = function(n){
28     this.n = n
29   }
30
31   /* set number of additional samples to be included for test phase */
32   this.set_m = function(m){
33
34     /* subsequently to drawing "n" items from the pool (without replacement),
35        a further "m" samples are drawn from the pool. For the test phase, the
36       "n" and "m" selections are mixed together and shuffled. */
37     this.m = m
38   }
39
40   /* get */
41   this.get_n = function(){
42     return this.n
43   }
44
45   /* get */
46   this.get_m = function(){
47     return this.m
48   }
49
50   /* remove any "blank" elements that appeared from drawing elements without
51      replacement */
52   this.remove_blanks = function(){
53     this.stimuli = this.stimuli.filter(function(){return true})
54   }
55
56   /* pseudorandom selection of size "n" */
57   this.draw_n = function(){
58
59     if(this.selection_n){
60       console.log('error: n-selection already made from this pool.')
61       return null
62     }
63
64     /* check the selection size */
65     var n = parseInt(this.get_n())
```

```
66      if (n > this.stimuli.length){
67        console.log('error: n > this.stimuli.length')
68        return null
69      }
70
71      /* make a pseudorandom selection */
72      this.selection_n = new Array()
73      var rem = this.stimuli.length
74      for(var i = 0; i < n; i++){
75        var qx = rand() * parseFloat(rem --), idx = parseInt(qx)
76        this.selection_n.push(this.stimuli[idx])
77        delete this.stimuli[idx]
78        this.remove_blanks()
79      }
80    }
81
82    /* pseudorandom selection of size "m" */
83    this.draw_m = function(){
84
85      if(this.selection_m){
86        console.log('error: m-selection already made from this pool.')
87        return null
88      }
89
90      /* check the selection size */
91      var m = parseInt(this.get_m())
92      if(m > this.stimuli.length){
93        console.log('error: m > this.stimuli.length')
94        return null
95      }
96
97      /* make a pseudorandom selection */
98      this.selection_m = new Array()
99      var rem = this.stimuli.length
100       for(var i = 0; i < m; i++){
101         var qx = rand() * parseFloat(rem --), idx = parseInt(qx)
102         this.selection_m.push(this.stimuli[idx])
103         delete this.stimuli[idx]
104         this.remove_blanks()
105       }
106     }
107
108   /* for initializing a test phase: mix "N"-selection and "M"-selection together */
109   this.reshuffle = function(){
110
111     /* put the "N"-selection and "M" selection, together in array to_shuffle,
112        which will be shuffled */
113     var to_shuffle = [], i = 0
114
115     /* add the "N"-selection */
116     for(i = 0; i < this.selection_n.length; i++){
117       var dat_i = new Array()
118       dat_i.push(this.selection_n[i])
119       dat_i.push(true)
120       to_shuffle.push(dat_i)
121     }
122
123     /* add the "M"-selection */
124     for(i = 0; i < this.selection_m.length; i++){
125       var dat_i = new Array()
126       dat_i.push(this.selection_m[i])
127       dat_i.push(false)
128       to_shuffle.push(dat_i)
129     }
130
131     /* "shuffle"-- randomize the ordering of the combined array */
132     var shuffled = new Array(), deja_vu = new Array(), rem = to_shuffle.length
133     while((rem --) > 0){
```

```
134        var idx = parseInt(rand() * parseFloat(rem)), dat_i = to_shuffle[idx]
135        shuffled.push(dat_i[0])
136        deja_vu.push(dat_i[1])
137        delete to_shuffle[idx]
138        to_shuffle = to_shuffle.filter(function(){return true})
139      }
140      return [shuffled, deja_vu]
141    }
142
143
144    /* perform all of the above */
145    this.draw = function(){
146      this.draw_n()
147      this.draw_m()
148      this.reshuffle()
149    }
150
151    /* set N, M parameters and make a selection cf the above */
152    this.select = function(n, m=n){
153      this.set_n(n)
154      this.set_m(m)
155      this.draw()
156    }
157
158    /* end of "pool::pool()" */
159    return this
160 }
161
162 /* following the convention to wrap away the new() operator */
163 function stimulus_pool(){
164    return new pool()
165 }
```

6.6. **state.js.**

```
1  /* global counter for states/ AKA frames/ AKA slides */
2  var last_state_id = -1
3
4  /* reference to 2d canvas graphics context */
5  function get_ctx(){
6    return canvas.getContext("2d") //document.getElementsByTagName("canvas")[0].getContext("2d");
7  }
8
9  /* state: generic object representing trial (like a card in "hypercard") */
10 function state(expiry_ms =      0,  /* max. presentation time (mS) */
11                key_expiry =  true,  /* force expiry by key-press (true <--> on) */
12                intvl_ms   =      0,  /* interval btwn stimuli.. (ISI) 'blank slide' */
13                img_idx    =     -1,  /* image data (if any) */
14                txt        =  null,  /* text data (if any) */
15                successor  =  null){
16   var ctx = get_ctx()
17   this.action = null, this.ding = false, this.id = ++ last_state_id
18
19   /* is a key-press required to transition? */
20   this.key_required = false
21
22   /* array to store admissible key-codes for data entry or transition to next "slide":
23     default: M, N */
24   this.admissible_keys = [77, 78]
25
26   this.get_admissible_keys = function(){
27     return this.admissible_keys
28   }
29
30   this.clear_admissible_keys = function(){
31     this.admissible_keys = new Array()
32   }
33
34   this.add_admissible_key = function(k){
35     this.admissible_keys.push(k)
36   }
37
38   /* this array will record the keystroke data received while residing in this state */
39   this.key_strokes = new Array()
40
41   this.record_key_stroke = function(k){
42     this.key_strokes.push(k)
43   }
44
45   this.set_pool_id = function(pid){
46     this.pool_id = pid
47   }
48
49   this.get_pool_id = function(){
50     return this.pool_id ? this.pool_id : ""
51   }
52
53   /* keep a reference to this state, if it's the first one ever.. */
54   if(ctx.first_new_state == null){
55     ctx.first_new_state = this
56   }
57
58   /* only applies if there's a "next" trial, if this is a trial */
59   this.intvl_ms = intvl_ms
60
61   /* numeric */
62   this.expiry_ms = expiry_ms
63
64   /* boolean */
65   this.key_expiry = key_expiry
66
```

```
67    /* global image index (images added as member of ctx) */
68    this.img_idx = img_idx, this.successor = null, this.predecessor = ctx.last_new_state
69
70    this.require_key = function(){
71      return this.key_required
72    }
73
74    var id = (this.predecessor == null) ? -1 : this.predecessor.id
75    ctx.last_new_state = this
76
77    /* sanity check: make sure the predecessor points here */
78    if(this.predecessor){
79      this.predecessor.set_successor(this)
80    }
81
82    /* where are we going? */
83    this.set_successor = function(s){
84      this.successor = s
85    }
86
87    /* plot text or images */
88    this.show = function(){
89
90      /* execute associated action, if we have one */
91      if(this.action){
92        this.action(this)
93      }
94      var ctx = get_ctx()
95      ctx.clearRect(0, 0, ctx.w(), ctx.h())
96
97      /* upper text */
98      if(this.txt){
99        wrap_text(this.txt, ctx, 0)
100     }
101
102     /* middle text */
103     if(this.txt2){
104       wrap_text(this.txt2, ctx, ctx.h() - (2 * ctx.font_size + 20))
105     }
106
107     /* img or middle text (if word stim) */
108     if(this.img_stim){
109       draw_img(this.img_stim, ctx)
110     }
111
112     /* might need the wrap_text back on for long strings.. */
113     if(this.wrd_stim){
114
115       /* no wrap */
116       centre_text(this.wrd_stim)
117     }
118
119     /* logo of no image/ lower text present */
120     if(!this.txt2){
121       ctx.draw_symbol()
122     }
123   }
124
125   /* state expires by timer or key press */
126   this.set_expiry = function(t_ms){
127
128     /* follow clock or key to keep the show going */
129     this.expiry_ms = t_ms
130
131     /* state expires by key press */
132     if(t_ms <= 0){
133       this.key_expiry = true
134     }
```

```
135    }
136
137    /* enter a state (begin) */
138    this.start = function(){
139      var ctx = get_ctx()
140
141      /* start the clock.. */
142      this.t0 = window.performance.now(), this.start_date_time = date_time()
143
144      /* do data dump, if we're at the end */
145      if(this.id >= last_state_id){ //== ctx.last_state){
146
147          /* window.location.href == http://domain/memory/examples/test_phase/memory.html */
148          var href = window.location.href
149
150          /* go through all the states and record (in string format) the info we'd like to appear on the
                 server */
151          var state_i = ctx.first_state, state_index = 0, message = "url,event_id,task_id,task_type,
                 trial_id,duration(mS),start(yyyy:mm:dd:hh:mn:ss:mls),end(yyyy:mm:dd:hh:mn:ss:mls),isi,set,
                 stim_type,stim_id,stim_pool_id,response\n"
152          for(var state_i = ctx.first_state; state_i != ctx.last_state; state_i = state_i.successor){
153
154            var stim_type = null, my_stim = null, pi = ""
155
156            /* "the right way to check if a variable is undefined or not" */
157            if(typeof state_i.pool_id !== 'undefined'){
158              pi = JSON.parse(JSON.stringify(state_i.pool_id))
159            }
160
161            /* assign "stimulus type" keyword */
162            if(state_i.wrd_stim){
163              stim_type = "word", my_stim = state_i.wrd_stim
164            }
165            if(state_i.img_stim){
166              stim_type = "image", my_stim = state_i.img_stim.fn
167            }
168            if(!stim_type){
169              stim_type = ""
170            }
171            if(!my_stim){
172              my_stim = ""
173            }
174
175            /* for a given "state", record a line of data */
176            message += href + ","
177
178            /* event_id: global index / line number */
179            message += state_index.toString() + ","
180
181            /* task_id */
182            message += state_i.task_id + ","
183
184            /* task_type */
185            message += state_i.type + ","
186
187            /* trial_id */
188            message += state_i.trial_id + ","
189            message += Math.round(10. * (state_i.t1 - state_i.t0)) / 10. + ","
190            message += parse_date_time(state_i.start_date_time).toString() + ","
191            message += parse_date_time(state_i.end_date_time).toString() + ","
192
193            /* ISI */
194            if(state_i.type == 'isi'){
195              message += state_i.expiry_ms.toString()
196            }
197            message += ","
198
199            if(!state_i.expiry_ms){
```

```
200              state_i.expiry_ms = ""
201          }
202
203          /* SET */
204          message += state_i.expiry_ms.toString() + ","
205
206          /* stimulus type */
207          message += stim_type.toString() + ","
208
209          /* stimulus id */
210          message += my_stim.toString() + ","
211
212          /* stimulus-pool id */
213          message += pi.toString() + ","
214
215          /* user response */
216          var response = '"'
217
218          if(state_i.type == 'delay'){
219
220            /* use the response text (not the sequence of characters). When testing with Max,
                    discovered we could see a symbol for each keystroke, in the data stream (incl., e.g.,
                    backspace characters). We want the final result, not the intermediary. */
221            response += state_i.txt
222          }else{
223
224            /* write out the individual response key(s) in terms of the representative characters */
225            for(var k in state_i.key_strokes){
226              response += String.fromCharCode(state_i.key_strokes[k])
227            }
228          }
229          message += response + '"'
230          if(response=='"""'){
231            response = ''
232          }
233
234          /* filter the response data for possible newline characters */
235          response.replace('\n', ' ')
236
237          /* add a newline character */
238          message += "\n"
239
240          /* go next */
241          ++ state_index
242        }
243
244        /* remove last three elements from array: take current page and navigate to:
245            ../../xml-receive.py == http://domain/memory/xml-receive.py */
246        var words = href.split('/')
247        var nwords = words.length
248        var target = words.splice(0, nwords-3).join('/') + '/xml-receive.py'
249
250        /* send the message to the server-side script at URL: target */
251        xml_send(message, target)
252    }
253
254    /* clear the timer */
255    ctx.clear_tmr()
256
257    /* plot the current trial */
258    this.show(ctx)
259
260    /* start the timer? */
261    if(this.expiry_ms > 0){
262      ctx.init_tmr(this.expiry_ms, this.expire)
263    }
264    return null
265  }
```

```
266
267    /* pr0c33d t0 th3 n3xt 5+4t3 */
268    this.expire = function(){
269      var ctx = get_ctx()
270
271      /* st0p 4ll th3 cl0ck5 */
272      ctx.clear_tmr()
273
274      /* r3c0rd st0p t1m3 */
275      this.end_date_time = date_time(), this.t1 = window.performance.now()
276      var txt = this.txt, suc_txt = null, suc = this.successor
277
278      if(suc && suc.txt){
279        suc_txt = suc.txt
280      }
281
282      /* enter next state */
283      if(this.successor && (this.successor!=this)){
284        ctx.set_state(this.successor)
285        ctx.get_state().start()
286      }
287    }
288    return this
289 }
```

6.7. **task.js.**

```
1  /* Event hierarchy: 1) Experiment (includes multiple tasks) 2) Task (includes multiple trials) 3) Trial
       (each task includes multiple basic events) */
2
3  /* instructions task (show a slide with a message on it) */
4  function instructions(txt){
5    var my_task_id = next_task_id++
6
7    /* initialize generic "trial" object */
8    var x = new state()
9
10   /* set associated text field */
11   x.txt = txt
12
13   /* no timer for the trial */
14   x.set_expiry(0)
15   x.type = 'instructions', x.task_id = my_task_id, x.trial_id = 0
16   return x
17 }
18
19 /* previously known as feedback task */
20 function feedback(txt, keys){
21   var my_task_id = next_task_id ++
22
23   var x = new state()
24   x.set_expiry(0)
25   x.txt = txt, x.key_required = true
26   x.clear_admissible_keys()
27   for(var i in keys){
28     x.add_admissible_key(keys[i])
29   }
30   x.type = 'feedback', x.trial_id = 0, x.task_id = my_task_id
31 }
32
33 /* list as many countries as possible during e.g., a 3-minute period (default, 30s)
34    20170515: default for delay_time used to be 30000. Today we added the end on <esc>
35    key feature
36 */
37 function delay_task(txt, delay_time=0, isi_=500){
38   var my_task_id = next_task_id ++, isi = parseInt(isi_)
39
40   /* if ISI was set, prefix with a "blank" slide */
41   if(isi > 0){
42     var x = new state()
43     x.set_expiry(isi)
44     x.type = 'isi', x.wrd_stim = "", x.trial_id = 0, x.task_id = my_task_id
45     x.clear_admissible_keys()
46     x.key_expiry = false
47   }
48
49   var y = instructions(txt)
50
51   /* time [mS] */
52   var x = new state()
53   x.set_expiry(delay_time)
54   x.key_expiry = false, x.txt = '', x.type = 'delay', x.trial_id = 0, x.task_id = my_task_id
55   if(delay_time <= 0){
56     x.clear_admissible_keys()
57     x.add_admissible_key(27)
58     console.log('admissible_keys', x.admissible_keys)
59   }
60   return x
61 }
62
63 /* study phase, formerly known as orientation task: multiple 'trials' / events occur here.. random
       selection of inputs... (for the test phase, the random selection is shuffled back into the pool)..
       */
```

```
64 function study_phase(my_pool, isi=0, time_limit=0, extra_feedback=false, extra_feedback_message="",
       extra_feedback_keys=[]){
65
66    /* the above constructor (same with test_phase) can accept either a single stimulus pool (pool()),
67       or an array of stimulus pools (pool()) */
68    var my_pools = []
69    if(my_pool.is_pool){
70      my_pools.push(my_pool)
71    }else{
72      my_pools = my_pool
73    }
74
75    var trial_index = -1, my_task_id = next_task_id++
76    this.ctx = ctx, this.p = my_pools, this.pool_ids = new Array()
77
78    /* for study phase, selection is built from combination of all selection_n arrays, from each of the
          supplied pools */
79    var my_selection = new Array()
80    for(var a_pool in my_pools){
81      var my_pool = my_pools[a_pool]
82      this.pool_ids.push(my_pool.pool_id)
83      for(var i in my_pool.selection_n){
84        var extra_feedback_this_slide = false
85        if(extra_feedback != false){
86          if(0 == i % parseInt(extra_feedback)){
87            extra_feedback_this_slide = true
88          }
89        }
90        my_selection.push([my_pool.selection_n[i], my_pool.pool_id, extra_feedback_this_slide])
91      }
92    }
93
94    /* randomize the order of the array */
95    shuffle(my_selection, true)
96
97    for(var selection_ind in my_selection){
98
99      /* increment the trial-index counter */
100     ++ trial_index
101
102     var a_selection = my_selection[selection_ind]
103
104     /* data (word or image) assigned to "trial" */
105     var data = a_selection[0], p_id = a_selection[1], extra_feedback_this_slide = a_selection[2]
106
107     /* if ISI was set, prefix with a "blank" slide */
108     if(isi > 0){
109       var x = new state()
110       x.set_expiry(isi)
111       x.type = 'isi', x.wrd_stim = "", x.trial_id = trial_index, x.task_id = my_task_id
112       x.set_pool_id(my_pool.pool_id)
113       x.clear_admissible_keys()
114       x.key_expiry = false
115     }
116
117     /* initialize generic "trial" object for each case */
118     var x = new state()
119     if(time_limit <= 0){
120       x.set_expiry(0)
121       x.key_required = false
122     }else{
123       x.set_expiry(time_limit)
124       x.key_required = false
125     }
126
127     /* discern by image or word, respectively */
128     if( typeof(data) === 'object'){
129       x.img_stim = data
```

```
130        }else if(typeof(data) === 'string'){
131          x.wrd_stim = data
132        }
133        x.type = 'study_phase', x.trial_id = trial_index, x.task_id = my_task_id
134        x.set_pool_id(p_id)
135        if(extra_feedback_this_slide){
136          var x_f = feedback(extra_feedback_message, extra_feedback_keys)
137        }
138      }
139      return this
140 }
141
142 /* test phase, formerly known as recognition task — for this phase,
143 the random selection is shuffled back into the pool — all elements
144 from the pool are shown (feedback is recorded).. */
145 function test_phase(my_pool, isi=0, time_limit=0, extra_feedback=false, extra_feedback_message="",
        extra_feedback_keys=[]){
146    var my_pools = []
147    if(my_pool.is_pool){
148      my_pools.push(my_pool)
149    }else{
150      my_pools = my_pool
151    }
152
153    var trial_index = -1, my_task_id = next_task_id++
154    this.ctx = ctx, this.p = my_pools, this.pool_ids = new Array()
155
156    /* for test phase, selection is built from combination of all selection_m arrays, from each of the
          supplied pools */
157    var my_selection = new Array()
158    for(var a_pool in my_pools){
159      var my_pool = my_pools[a_pool]
160      this.pool_ids.push(my_pool.pool_id)
161      var trial_index = -1, shuffled_data = my_pool.reshuffle(), shuffled = shuffled_data[0], deja_vu =
            shuffled_data[1]
162      for(var i  in shuffled){
163        var extra_feedback_this_slide = false
164        if(extra_feedback != false){
165          if(0 === i % parseInt(extra_feedback)){
166            extra_feedback_this_slide = true
167          }
168        }
169        my_selection.push([shuffled[i], my_pool.pool_id, deja_vu[i], extra_feedback_this_slide])
170      }
171    }
172    shuffle(my_selection, true)
173
174    for(var selection_ind in my_selection){
175      ++ trial_index
176
177      var a_selection = my_selection[selection_ind]
178      var data = a_selection[0], p_id = a_selection[1], deja = a_selection[2], extra_feedback_this_slide
            = a_selection[3]
179
180      /* if ISI was set, prefix with a "blank" slide */
181      if(isi > 0){
182        var x = new state()
183        x.set_expiry(isi)
184        x.type = 'isi', x.wrd_stim = "", x.trial_id = trial_index, x.task_id = my_task_id
185        x.set_pool_id(p_id)
186        x.clear_admissible_keys()
187        x.key_expiry = false
188      }
189
190      var x = new state()
191      x.key_required = true
192      if(time_limit <= 0){
193        x.set_expiry(0)
```

```
194        }else{
195          x.set_expiry(time_limit)
196        }
197
198        /* record within the object: do we have deja-vu? */
199        x.deja = deja
200
201        /* word or image? */
202        if( typeof(data) === 'object'){
203          x.img_stim = data
204        }else if(typeof(data) ==='string'){
205          x.wrd_stim = data
206        }
207        x.type = 'test_phase', x.trial_id = trial_index, x.task_id = my_task_id
208        x.set_pool_id(p_id)
209
210        if(extra_feedback_this_slide){
211          var x_f = feedback(extra_feedback_message, extra_feedback_keys)
212        }
213      }
214    var m = 'Thank you for completing this section. ', end = instructions(m)
215
216    end.action = function(me){
217        var msg = m + 'Your score: ' + ctx.questions_correct.toString() + '/' + ctx.questions_total.
                toString() + ". Please press any key."
218        me.txt = msg
219      }
220    return this
221 }
```

6.8. **text.js.**

```js
/* wrap text around a window region— via ashblue */
function wrap_text(s, ctx, start_y=0){
  var myX = 10, myY = 50, line = '', lines = [], w = ctx.w(), h = ctx.h(), line_test = '', words0 = s.
      split(' '), font_size = ctx.font_size
  ctx.font = font_size +'px Arial'
  var words = new Array()
  for(var i = 0; i < words0.length; i++ ){
    var w = words0[i]
    ws = w.split('\n')
    words.push(ws[0])
    if(ws.length > 1){
      console.log("ws", ws)
      for(var j = 1; j < ws.length; j++){
        words.push('\n')
        if(ws[j] != ""){
          words.push(ws[j])
        }
      }
    }
  }

  w = ctx.w()

  /* place words one by one */
  for(var j = 0; j < words.length; j++){
      if(words[j] == "\n"){
        myY = lines.length * font_size + font_size
        lines.push({text: line, height: myY})
        line = ''
        continue
      }

      line_test = line + words[j] + ' '

      /* wrap if over the edge */
      if(ctx.measureText(line_test).width > w){
        myY = lines.length * font_size + font_size
        lines.push({text: line, height: myY})
        line = words[j] + ' '
      }else{
        line = line_test
      }

  }

  /* catch last line if something left over */
  if(line.length > 0){
    current_y = lines.length * font_size + font_size
    lines.push({text: line.trim(), height: current_y})
  }

  /* plot text */
  for(var j = 0, len = lines.length; j < len; j++){
    ctx.fillText(lines[j].text, 0, lines[j].height + start_y)
  }
}

/* write centred text */
function centre_text(s){
  var font_size = ctx.font_size, textString = s
  ctx.font = 30 + 'px Arial'
  textWidth = ctx.measureText(textString).width
  ctx.fillText(textString, (canvas.width / 2) − (textWidth / 2), canvas.height / 2)
}
```

6.9. **util.js.**

```
1  /* cr34t3 a c4nv4s wh3r3 th3 m4g1c h4pp3ns */
2  var canvas = document.createElement('canvas')
3  document.body.appendChild(canvas)
4
5  /* get date and time */
6  function date_time(){
7    return new Date()
8  }
9
10 /* seed for rand() below */
11 var seed = 5
12
13 var get_seconds = function(){
14   var d = new Date()
15
16   /* return an epoch time (S) */
17   return d.getMilliseconds()
18 }
19
20 var mutable_seed = get_seconds()
21
22 /*random-number generator http://indiegamr.com/generate-repeatable-random-numbers-in-js/ : initial seed
      .. in order to work 'Math.seed' must NOT be undefined, so in any case, you HAVE to provide a Math.
      seed */
23 function rand(max, min, mutable=false){
24     max = max || 1, min = min || 0
25   if(mutable){
26     mutable_seed = (mutable_seed * 9301 + 49297) % 233280
27     return min + (mutable_seed / 233280) * (max - min)
28   }else{
29     seed = (seed * 9301 + 49297) % 233280
30     return min + (seed / 233280) * (max - min)
31   }
32 }
33
34 /* Shuffle array in place, via http://stackoverflow.com/questions/6274339/how-can-i-shuffle-an-array
35  * @param {Array} a items The array containing the items.
36
37   setting the parameter "mutable" to true, makes random selections that will change between runs. */
38 function shuffle(a, mutable=false) {
39   var j, x, i
40   for(i = a.length; i; i--){
41
42     /* use our seeded random number generator, so we get the same results every time */
43     j = Math.floor(rand(null, null, mutable) * (1. * i))   /* j = Math.floor(Math.random() * i) */
44     x = a[i - 1]
45     a[i - 1] = a[j]
46     a[j] = x
47   }
48 }
49
50 /* pad to length n (with 0's on the left) */
51 function pad_n(x, n){
52   var s = parseInt(trim(x)).toString(), m = s.length, d = n - m
53   if(d > 0){
54     s += '0'.repeat(d)
55   }
56   return s
57 }
58
59 /* via stackoverflow.com/users/4321/jw */
60 function get_keys(dictionary){
61
62   /* keys recursive */
63   var keys = []
64
```

```
65    /* filter for direct ancestors */
66    for(var key in dictionary){
67      if(dictionary.hasOwnProperty(key)){
68        keys.push(key)
69      }
70    }
71    return keys
72 }
73
74 /* draw an image */
75 function draw_img(x, ctx){
76    var cf = 4 * ctx.font_size
77    var h = ctx.h() - cf, w = ctx.w()
78    var lw = x.width, lh = x.height
79    var sf = Math.min(w, h) / Math.max(lw, lh)
80    var a = (w - lw * sf) / 2, b = (h - lh * sf) / 2
81    var c = lw * sf, d = lh * sf, df = (-20 + cf / 2)
82    ctx.drawImage(x, a, b + df, c, d)
83 }
84
85 /* write the above to a standardized format */
86 function parse_date_time(today){
87
88    /* most significant units first */
89    var bits = [today.getFullYear(),
90                today.getMonth() + 1,
91                today.getDate(),
92                today.getHours(),
93                today.getMinutes(),
94                today.getSeconds(),
95                today.getMilliseconds()]
96
97    /* pad with zeros */
98    for(var i = 0; i < bits.length; i++){
99      var n_pad = 2
100     if(i == 0){
101       n_pad = 4
102     }
103     if(i == 6){
104       n_pad = 3
105     }
106     var bts = bits[i].toString()
107     bits[i] = pad_n(bts, n_pad)
108   }
109   return(bits.join(':'))
110 }
111
112 /* "faster trim" via blog.stevenlevithan.com */
113 function trim(s){
114   return s.toString().replace(/^\s\s*/,'').replace(/\s\s*$/,'')
115 }
116
117 /* send text format data (string s) via XML to receive script at url (string): xml-receive_script_url
       */
118 function xml_send(s, xml_receive_script_url){
119
120   /* xml http request object */
121   var xhr = (window.XMLHttpRequest) ? new XMLHttpRequest() : new activeXObject("Microsoft.XMLHTTP")
122   var data = new FormData()
123   data.append("data", s)
124   xhr.open('post', xml_receive_script_url, true)
125   xhr.send(data)
126 }
```

7. Source Code: Server Side

The folder **data/** in the directory structure, relative to the installation folder: if it doesn't yet exist, the server-side python code will create it.

7.1. **xml-receive.py.**

```python
#!/usr/bin/python
''' server-side python-CGI script to receive text data sent over
the internet by the client-side function util.js::xml_send() '''
import os
import cgi
import uuid
import datetime

# create /data folder if it does not yet exist
dat_f = os.getcwd() + '/data/'
if not os.path.exists(dat_f):
    os.mkdir(dat_f)

# retrieve CGI form data
dat = None
try:
    dat = str(cgi.FieldStorage().getvalue('data'))
except:
    pass

# write the data to file in the data/ folder
if dat:
    fn = dat_f + str(datetime.datetime.now().isoformat())
    open(fn + '_' + str(uuid.uuid4().hex) + '.txt', 'wb').write(dat)
```

8. Recommendations For Further Improvements

Here's a short point-form list of possible improvements to the software:

- Finish drag-and drop implementation, that
  - prevents programmatic errors;
  - does not allow invalid experiments to be constructed; and
  - removes any technicality from the process of coding an experiment.
- Smarter image loading
  - Automagically detect available images from folder, and
  - modify program to allow administrator to upload images with file names that aren't required to follow the numbered format.
- Support for mobile devices, or possibly:
  - show a warning message for un-supported devices.
- Accomplish API-level integration with "Mechanical Turk" for detailed/complex systematic/automated deployment of surveys for use in Recognition Memory experiments.