“MSA runner” app - specs

# Basic features

The participant sees one or more lines of code at a time, and needs to type in the response to each line.

## Terminology

**Stimulus:** something that appears on screen in a given time point (one or more lines of code).

**Response:** what the participant types (on keyboard)

**Step:** a combination of a single stimulus and a single response

**Program / trial:** a set of lines that are conceptually a single program.

**Session:** The whole experiment session, with several program.

## Responses

* Steps without response: the participant only types SPACE
* Steps with response:

Numerical responses: 1-9, “0” (means 10), the keys Q-P (= 11-20)

TAB means “<1”

Closing bracket (]) means “>20”

After the user clicks any response, a feedback sound is played.

## Stimuli

**When we’re in a step:**

1. Lines appear in the middle of the screen. If there are multiple lines of code (e.g. initializing several variables), they appear one below the other.
2. In “no response trials”, the text “>>>>>>” appears at the bottom of the screen.
3. Progress bar at the top of the screen.
4. Nothing else appears on screen.

**Between programs**

* A fixed message
* The “>>>>>>” text at the bottom
* A grey text at the bottom-right, indicating the program number.

## Configuration

The programs appear in an input XLS/CSV file (whichever you prefer). See the attached config1.xlsx, I marked in red the columns with functional relevance.

* The column “trial\_code” is the program ID. i.e. when it changes, it means a new program has started. This is the ID that appears on the screen between programs.
* The column “response needed” indicates whether the step requires a response or just clicking SPACE.

## Output file

The output CSV should contain all the columns from the input file, plus columns generated by jspsych:

* *rt*: the reaction time (how long it took to respond)
* *response*: the response key pressed by the participant
* *response\_value*: the numerical value that this key represents, or the words “small” (for TAB) or “large” (for “[“).
* Another column (I forgot its name) which is the time since the session started.

# Functions

Now the code may contain functions. The text of main appears on the top half of the screen, the text of the function at the bottom. There’s a horizontal line in the middle. Nothing else changes.

* The config file will now contain a “text2” column. In each step, text1 indicates what to show on top, text2 on bottom.
* The script should auto-detect whether the config file contains a text2 column or not. If yes – run in functions mode. If no – run as described above in Section 1.

See the attached file config\_funcs.xlsx

# Loops

The code may contain loops (see config-loops.xlsx)

Some steps are loop, some are not. This is defined by the “is\_loop” column in the config file.

If the step is not a loop, everything behaves as described above. If the step is a loop, the following changes occur:

* When the participant clicks a response, we don’t proceed to the next step but keep showing the same screen.
* We proceed when the participant clicks “end loop” (backspace key).
* Each loop will result in several lines in the CSV output file (including the click on “end loop”, which is also a line). They will all have the same step number, but please add a “loop step” column, which is restarted to 1 in each new loop (and then 2, 3, etc.).

# Feedback

Feedback mode is activated via a URL parameter “feedback=1”.

In this mode, the participant gets feedback whenever they make a mistake.

Note that in loops mode,

* The expected\_response column contains several values, in accordance with the number of expected responses during this loop.
* The script should implicitly expect a BACKSPACE response at the end of the loop.

**How to show the feedback:**

* The format of the feedback text is always of the form “x=#”: “x” is the variable that was just updated (take from updated\_var column in the config file), and “#” is the correct response (take from expected\_response column in the config file).

Exception: when we expected the user to hit BACKSPACE (end of loop) and he didn’t, the feedback will be just the text “loop ended”.

Under this text, print “Hit space to continue”.

* The feedback text appears in a different color (an easy-to-read green).
* When feedback is needed following an error, don’t play the standard “click” but a different sound (error.wav).
* After feedback was shown, the user needs to click space to continue to the next step (which may be either the same stimulus if we’re still in the loop, or a new stimulus).