HowFast project – keyseq experiment

Initial experiment to test for the validity of the typing time measurement

F.-X Alario[1], S. Dufau[1], M. Longcamp[2], S. Mathôt[1], S. Pinet[1], C. Zielinski[3]
[1]Laboratoire de Psychologie Cognitive ; [2]Laboratoire de Neuroscience Cognitive ; [3]Brain and Language Research Institute

1- Development of the on-line experiment

1.1- General description

The online experiment consisted of a set of HTML, JavaScript, CSS and PHP files as described in figure 1.

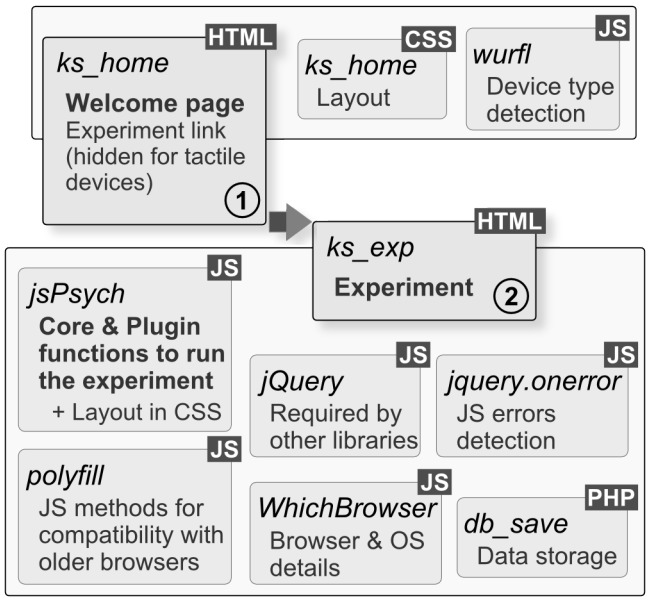

Figure 1: main files and libraies used for the online experiment

These files were stored on the disk of a server machine that was already set up and managed by Sebaastian Mathôt (coauthor of this study) - and more precisely associated with *cogsci.nl* domain name. The collected data were saved on the mySQL database that was configured on this server machine.

The welcome HTML page (*ks_home*) allowed explaining briefly to the participant the context of the study, the general principle of the experiment and its duration. It contained the link to launch the experiment. As the task involved keystrokes, this link was only visible to computer-based browser (wurfl tool), excluding participants using tablets or smartphones.

The whole experiment process took place in the second HTML file (*ks_exp*), that was opening in a lightened full size window (neither header menu nor scroll bars) by clicking on the link.

The layout and formatting information, like the locations of the paragraphs and images, or the names, sizes and colors of the fonts, were indicated in the CSS files. JavaScript tools were embedded in the HTML files for specific needs, such as detecting the king of device used by the participant (wurfl.js), and

for programming the experiment itself (jsPsych library). The PHP files provided the instructions to access to the database and to store the data.

The complete list of the tools that were involved in our experiment is shown in the table X.

1.2 – JavaScript for running behavioral experiments

To display the stimuli and capture the reactions times online, we opted for the JavaScript language combined with HTML and CSS. We used in particular the JavaScript library jsPsych to program the experiment **[1]** (see 1.4).

JavaScript is natively supported by all modern browsers, such as Firefox or Chrome. It does not require any installation or updating of browser plugins. It is interpreted and run locally by the browser, i.e. on the client-side. This ensures a responsive experimental design and an accurate measurement of response times, as it is not affected by remote server and network latencies.

However, some not up-to-date browsers could not be able to interpret recent JavaScript functions. Nevertheless, this problem can be avoided by including an additional script to the header of the HTML document. This script is called a polyfill. It contains code that defines the methods that might be missing in old browser versions.

1.3 - Basics on JavaScript

JavaScript makes the content of the page to be interactive and animated. It allows modifying HTML code on the fly, by inserting and deleting elements (e.g. text, images), or by changing elements properties. It holds predefined methods to manage and create HTML elements. It is also a complete programming language, allowing to manipulate variables like text or number arrays, to make mathematical operations and finally to define its own functions.

At the opening of the HTML page, all the embedded JavaScript functions are loaded and ready to act on page elements. The execution of each function occurs in response to specific user action (mouse, keyboard event), or according to a predefined timeline.

1.4 – Use of the jsPsych library and new plugin implementation

The experiment was mainly developed from the open-source **jsPsych library (v4.3 2015) [1]**. This library is designed to build behavioral experiments with a minimum of programming skills. It holds predefined methods to manage the experiment timeline, to collect reaction times and user actions (mouse, keyboard events), as well as to randomize stimuli, handle the data, and prepare data for the backup.

All these features are embedded as JavaScript functions in a main script called the core script. To this script are associated plugin functions that are specific to the stimulation.

Each plugin describes the process for one trial. It comes with a set of input parameters such as the display time duration, the expected response keys and the inter-stimulus duration, as well as the list of stimulus material to display (e.g. paths of image files, array of words or symbols).

Plugin execution outputs a JavaScript object containing the recorded events and reaction times, which is then added to the current global data set.

Various plugins are available in jsPsych library, which enabled the execution of common behavioral tasks such responding as quickly as possible to a stimulus by pressing a key or sorting images by clicking and dragging. For specific needs, new plugins can easily be conceived, by adapting an existing one or by starting from a template file.

Two plugins were hence written for the current experiment. The first one was devoted to the main task of the study, involving the typing of a 3-keys sequence as a response to a visual stimulus (i.e. a character appearing on the screen). This plugin, called *jspsych-key-sequence*, allowed to record the typing times of each pressed key and to display a visual feedback by changing the color of the stimulus before it disappeared (at the end of the 3nd typing, the initially black symbol became green if the key-sequence was correct, red otherwise). The visual stimulus, the corresponding expected typing sequences, the total number of stimulus and the inter-stimulus duration were configurable as input parameters. When given several types of stimuli and associated sequences, the stimuli were firstly randomly shuffled before to be displayed successively. The plugin also offered a "training mode" that could be switched on from the input settings. In that case, a word was adding below the display symbol, in addition to the feedback color indication ("REUSSI" *(succeeded)* associated with the green symbol and "RATE" *(missed)* associated with the red one). This mode was also causing the early end of the block, as soon as the keystroke sequence was wrongly typed (with the aim to remind the instructions as often as necessary to the participant during the training session). The parameters used for the experiment are detailed in the section xx.

In order to collect some relevant information about the participant during the experiment (like his age or handedness), the plugin *jspsych-form* was set as an improvement of the available *jspsych-survey-text* plugin. This latter was developed to collect only free answers through text input field. The new *jspsych-form* made it possible to display either free text area or clickable predefined options (e.g. left-, right-handed or ambidextrous) in association with any question of the form. It also allowed customizing the width of each input text area of the form, as input settings.

To use jsPsych, the paths of the jsPsych core script and those of the needed plugins have to be firstly indicated in the header section of the HTML experiment file (as well as the CSS file and the jQuery script which is required by jsPsych). In an underneath JavaScript section, the experimenter defines the whole structure of the experiment as a succession of blocks of trials, by invoking plugins with their set of

specific parameters and stimulation material lists. The data storage process can be done as a last jsPsych process by calling the required PHP files.

1.5 - Data storage

At the end of the experiment, data were transferring to the server using an Ajax request (*Asynchronous JavaScript and XML)*. The transferred data included the subject identifier, the OS/browser information and the jsPsych data, each as character array. The subject identifier was based on the accurate experiment starting date (to the millisecond). The OS/browser information was provided by the WichBrowser tool (Tab. 1). Data related to the experimental task comprised keys of sequences, associated typing times and survey answers. They were recorded as a JavaScript object during the experiment, that was subsequently transformed in plain text (i.e. stringified) before the sending to the server.

The transferred data set was finally stored in the MySQL database by the help of PHP files. The access to the database and the writing of the data inside the data table were done using PHP Data Object (PDO) extension, which ensured database portability and security.

1.6 - Debugging stage

The experiment was firstly tested internally through a set of recent operating systems (Windows, MacOS and Linux), and up-to-date common browsers (Mozilla Firefox, Chrome, Safari, Opera and Internet Explorer).

A pre-test including 58 participants was secondly done by sending the link to the experiment to the members of the associated laboratories.

To identify and overcome potential bugs, a spy library (Tab. 1) had been included in the experiment HTML file (which was only working with Firefox, Chrome or IE). As soon as an error was encountered by the participant's browser, the error message was automatically sent by an e-mail addressed to the developer. This message comprised the file path, code line and error in question. A message was also appearing on the participant's page to invite him to try again later due to an unexpected error.

A total of 7 errors were hence forwarded by the spy. Five of them were due to the use of old browser versions that were not able to interpret some JavaScript methods. To avoid these errors again, a polyfill had immediately been added in the experiment file. The other two errors were linked to the syntax that was used to set the properties of the HTML elements, not readable by the less recent browser. This was simply fixed by modifying the way of writing the properties.

No JavaScript error was reported by the spy thereafter during the main test.

The pre-test was also used to check for the understanding of the instructions.

2 – Online experiment

2.1 – Stimuli and sequences

The experimental task involved pressing 3 consecutive keys as a response to the apparition of a symbol on the screen. At each block of trials, two different sequences were associated with the "X" or "O" visual stimulus, stimuli appearing successively in a random order.

Following the choice of stimuli from the original experiment by Rosenbaum **[2]**, the possible sequences involved the index, middle and ring of the two hands.

Constant sequence implied fingers of the same hand with this typing order: index, ring, middle. It was intermixed with one of two Uncertain sequences, introducing the index of the other hand at the second or third position compare to the original Constant sequence.

For the entire experimental task duration, the participant's fingers were put on the same keys, ready to press the sequence each time the stimulus was appearing. The index, middle and ring fingers of the left hand were positioned respectively on the F, D and S keyboard letters, and those of the right hand on the J, K and L keys. These keys are located on the same place for Azerty and Qwerty keyboards, with the same spacing between the left and right letter sets.

At the launch of the experiment page by the participant's browser, the hand devoted to the Constant sequence (i.e. left or right) was drawn, as well as the associated visual stimulus ("X" or "O").

The specific plugin jspsych-key-sequence was called for each block of trials (see 1.2). The symbol was displayed in black color on the white background with a font size of 72 px. It was horizontally centered on the screen, at a height from the bottom of 63% relative to the total height of the page. The feedback duration (change of the symbol color) lasted for 650 ms. Inter-stimulus interval was set to 500 ms.

2.2 – Experimental design

The experiment was divided into two general parts (A and B). One of the two stimulus-sequence associations was fixed for the two parts, while the other was changing between the parts. The fixed sequence corresponded to the Constant sequence (see 2.1), which was associated with one of the two Uncertain sequences at each part. The assignment of the Uncertain sequences with the part A or B was done randomly for each participant.

At the start of the experiment, general instructions were given to the participant to show him how to place his fingers on the front keyboard. This was done by the help of the drawing of the two hands with keyboard letters written on the fingers. The pairing between visual stimuli ("X" and "O") and sequences was then introduced at the beginning of each part. Each sequence was defined by numbers (1 to 3) that were placed on the involved finger on a drawing of the hands.

Each part comprised a training phase followed by two test phases.

The training phase allowed the participants familiarizing with the sequences. It ended only when each sequence was correctly performed 4 times (the total 8 stimuli being randomly displayed). Each time an error was done, the training was reinitialized. Before it was resumed (by pressing the space bar), the paring instructions were showing again.

During each following test phase, a total number of 20 symbols (10 "X" and 10 "O") were appearing successively in a random order.

To keep subject motivation, a feedback page was appearing at the end of each test phase. The percent of correct answers and the mean speed of sequence typing were showed for the finishing phase and for the previous ones.

The experiment ended with a short survey, where participants were asked to answer a few questions (handedness, gender, age, employed by university or else) and had the opportunity to report if any problem occurred during the experiment. Figure 2 showed the general sketch of the design.

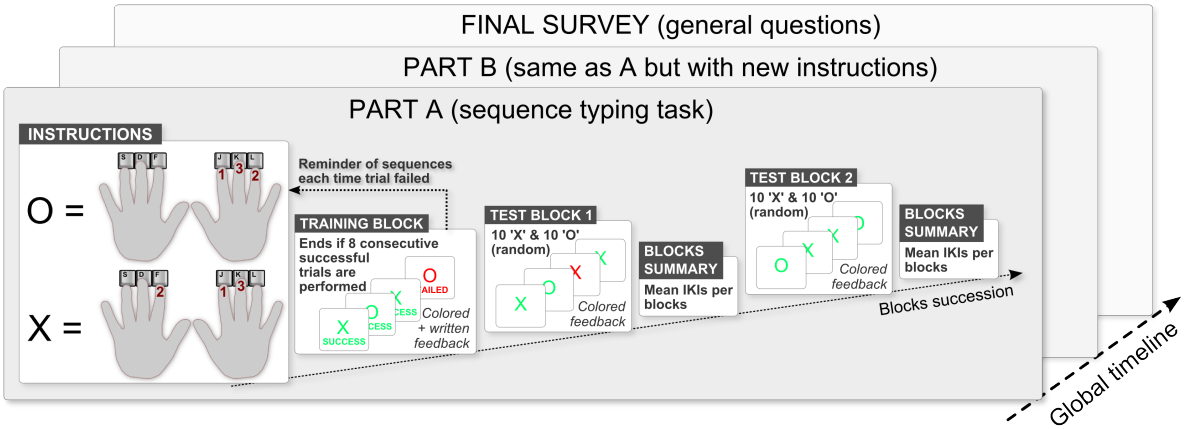

Figure 2: experimental design.

2.3 - Procedure

The online experiment was advertised through an email sent to all the Aix-Marseille university employees (more than 7500 people), by indicating the link to the welcome page on the server (ks_home.html). There was no monetary compensation offered.

Références

[1] de Leeuw, J. R. (2014). jsPsych: A JavaScript library for creating behavioral experiments in a Web browser. Behavior research methods, 1-12.

[2] D. A. Rosenbaum, Inhoff A. W., and Gordon A.M. (1983). Choosing Between Movement Sequences: A Hierarchical Editor Model. J. of Experimental Psychology, Vol. 113, No. 3. 372-393

Table 1 : References and descriptions of the additional tools embedded in the HTML files of the experiment

Tool name	Container files	Description	Reference / Link
wurfl.js	ks_home	Device detection – A simple way to know if the user device is a mobile, tablet or computer at the loading of the page	https://web.wurfl.io
jquery.onerror.js	ks_home; ks_exp	Error capturing plugin - Send an email when a JavaScript error is encountered by the user's browser, containing the error message	https://github.com/posabsolute/jQuery-Error-Handler-Plugin
jsPsych	ks_exp	Experiment development - Use of jspsych.js and of the jspsych-text.js plugin to display instruction + implementation of the 2 new plugins : jspsych-key-sequence.js (main task) and jspsych-form.js (survey)	http://www.jspsych.org (deLeeuw, 2014)
jquery.js	ks_home; ks_exp	JavaScript framework to simplify the client-side scripting of HTML - Required by jsPsych library and by jquery.onerror.js	http://jquery.com
polyfill.js	ks_exp	Additional code to overcome JavaScript bugs encountered by not up-to-date browser, regarding errors sent by jquery.onerror (including 4 functions that was copied from https://developer.mozilla.org/fr/)	
WhichBrowser	ks_exp	Library to access user's browser information (name, version, OS) from a simple JavaScript object	http://whichbrowser.net

List of useful software:

Open-source Notepad++ to write scripts ; Open-source ftp client FileZilla to drop files on the server ; Firebug plugin on Mozilla Firefox to test and debug programs