CONTENTS

Testable is a web-based platform that makes it possible to create, run, and share behavioural experiments quickly and easily. It doesn't require any programming and it's fast! Even experienced programmers may find it more convenient to use Testable than to write customised scripts to create standard experiments.

Despite its ease of use, Testable is comprehensive and versatile. It can display images and words, play sounds and movies, and create surveys. The response options include mouse and button clicks, key press and text input. All the standard experimental parameters, such as presentation time, inter-trial interval (ITI), inter-stimulus interval (ISI), are customisable. The platform allows for randomisation within and between blocks.

Testable is accessible at www.testable.org.

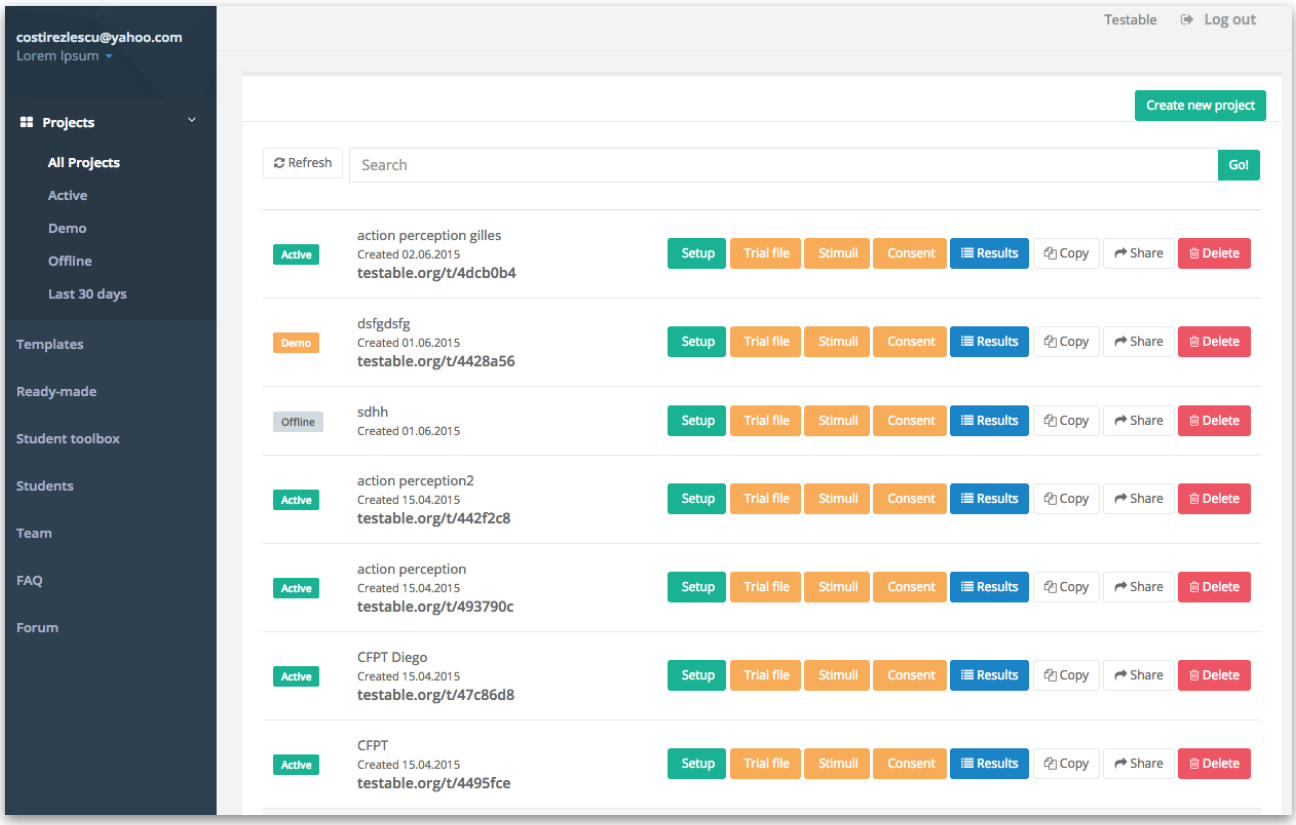
IMPORTANT: The platform is designed to display experiments on a canvas with width 1200px and height 800px. Images will be displayed in their original size, so make sure to upload images in the desired sizes. Also, make sure the images fit within the canvas, e.g. if you want to display three images, make sure each image's width is less than 400px (350px preferably) to be displayed correctly on the canvas.

How to create a new account and login

Click on the blue "LOGIN" button in the top right-hand corner of the screen. If you already have an account, you can enter your email address and password to login. If you don't have an account, sign up by providing the requested information.

NOTE: We recommend using your academic email account in case there is (will be) a general agreement with your institution that grants certain privileges to its staff and students.

After login, you should see the main project dashboard that looks like this (if you don't have any projects, you won't see a list of projects):



How to create a new Smart project

The easiest way to get started is to click on “Create new Smart project”. This will open a popup window (see image below) where you need to specify the type of experiment you want to have. When you are happy with the trials description, click DONE.

You can add multiple paragraphs describing the types of trials you want, including instructions screens. Once you finish describing all trial types you want, click on Generate project.

You will now have a new project called Smart Project in your projects list. If you plan to show images, sounds or videos, don’t forget to upload them in Stimuli and update the trial information spreadsheet (Trial form) with the corresponding filenames.

The new experiment if offline by default. You can make it live by changing its status from ”offline” to “active”. A link will be generated and you can click on it to check it.

How to create a new project

1. Click on the green button in the top right-hand corner of the screen called “Create new project”. That will open a page with setup information for the project (see **How to select the setup options, p. XX**)

2. Create a .csv spreadsheet with information about the experiment (see **How to fill in the spreadsheet, next**)

3. Upload the spreadsheet, stimuli, and consent (if required), by clicking on the corresponding buttons

4. Change the status of the test from ”offline” to “active”. A link will be generated. The experiment is now live (click on the link to verify).

How to select the setup options

You can choose to customise your experiment here, or you can accept all default options. The only required field is the Project Name. Most options are self-explanatory, so we will detail only a few of them. A screenshot of the Setup screen is presented on the next page.

Password protected

- you can protect your experiment so that only participants who know the password can access the experiment (you will be prompted to select a password)

START SCREENS: Calibration

- check this if you would like your stimuli to scale with the screen resolution of the participants' screens

START SCREENS: Consent

- check this if you want to obtain consent from participants before starting the experiment. If checked, you must upload the consent form as an image file (jpg, png, bmp)

START SCREENS: Participant details

- check this if you want to record participant's details. You will be asked whether you wish to allow participants under 18 to participate in the experiment.

START SCREENS: General instructions

- this option gives participants instructions about how to maximise the screen, avoid distractions etc.

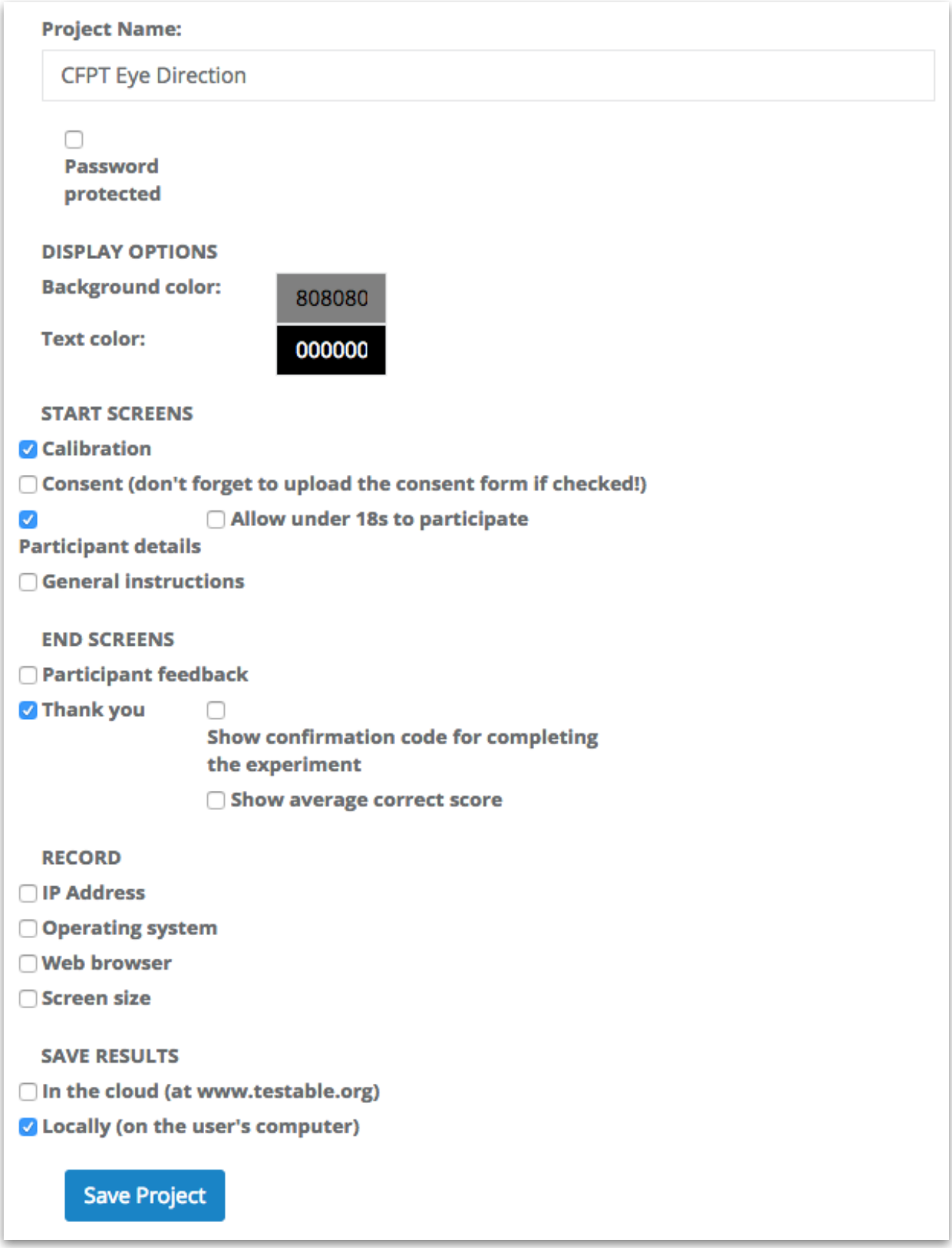
END SCREENS: Participant feedback

- this option allows participants to give you feedback/comments about the experiment

END SCREENS: Thank you

- this is a standard end screen, thanking participants for their time. It can include information about the average correct score on (selected) test trials and a confirmation code for completing the experiment (e.g. needed for experiments run on Amazon Mechanical Turks).

More options may be added in the future.


Project Name:
CFPT Eye Direction
Password protected
DISPLAY OPTIONS
Background color:
808080
Text color:
000000
START SCREENS
Calibration
Consent (don't forget to upload the consent form if checked!)
Allow under 18s to participate
Participant details
General instructions
END SCREENS
Participant feedback
Thank you
Show confirmation code for completing the experiment
Show average correct score
RECORD
IP Address
Operating system
Web browser
Screen size
SAVE RESULTS
In the cloud (at www.testable.org)
Locally (on the user's computer)
Save Project

How to fill in the spreadsheet

Knowing how to fill in the spreadsheet is the closest you will get to programming while using this platform. There are two principles:

1) Each row is a trial. Note that the term “trial” is used in a broad sense; it refers to experimental trials, but also instructions screens.
2) Each column is a parameter. The columns have pretty self-explanatory titles. You can delete columns/parameter you don’t need (with the exception of the column “type” which is required). You can add any column/parameter described in this manual. In addition, you can invent new columns/parameters to help you organize your experimental design and your results (these columns will not impact what participants see, but will be recorded in the results file - NOT IMPLEMENTED YET). The order of the columns is not fixed and columns can be reordered in any way.

You can download a template spreadsheet with many different types of trials **here**.

Below is a description of each column and the possible options/values.

BASIC COLUMNS

type {*instructions*, *test*, *practice*, *learn*}

This is the only required column. It determines what kind of trial should be presented. Possible values are:

instructions
- used to display instructions. Testable will display the text presented under **title** (in larger bold fonts) and **content** (in normal fonts).

test
- used for most experimental trials in which you want to present images, words, sounds, clips, surveys, and record responses from participants. Advancing to next trial is done after recording these responses. Testable expects at least one stimulus to be specified under **stim1**

practice
- same as ***trials***, except that participant responses are not recorded and will not appear in the data output. Responses are still required to advance to the next trial.

learn
- used for experimental trials with passive viewing. Testable will advance to the next trial after the time specified under **presTime**

form
- used to display forms (surveys).

title {*any text*}
content {*any text*}
- type the text you want displayed in the “instructions” trials.

stimFormat {*.jpg, .jpeg, .png, .gif, .bmp, .mp3, .mp4, word, survey*}
- specify the format of the stimuli to be displayed. The stimuli can be images (jpg, jpeg, png, gif, bmp), sounds (mp3), video clips (mp4), words (word) or the text response options for surveys (survey)

target {*filename*}
stim1 {*filename*}
stim2 {*filename*}
stim3 {*filename*}
stim4 {*filename*}
stim5 {*filename*}
stim6 {*filename*}
- specify the stimuli (with or without a target) you want to display in a trial
- test stimuli (sstim1…stim6) will be centered horizontally and vertically, and Testable will attempt to display the target above the test stimuli. If that is not possible because images are too large, everything will be shifted down so that target appears fully on the screen
- note that multiple audio and video files will be presented sequentially.
- filename should exclude the file extension. File extension is specified in stimFormat (p. 5)

stimPos {}
- stimuli are centered horizontally and vertically by default, but you can use this column to position them manually
- for each stimulus, you can define the horizontal and vertical displacement from a central position, eg. -50 50 means you want the stimulus 50px to the left and 50px down from a central position.
- if you have more stimuli, separate the sets of values by semicolon, eg. stimPos=-50 50;-100 100;-200 200 defines manual positions for three stimuli
- you can also specify a set or a range of options to select randomly from, eg. stimPos=10|20|30 100_200 will present the stimulus 10 or 20 or 30px to the left (selected randomly) and a random number between 100 and 200px down from a central position
- when using a range for random selection, you can also specify the step, eg. stimPos=0_100_20 will select a random number between 0 and 100 in steps of 20 (ie. 0, 20, 40, 60, 80 or 100)
- if you want a stimulus to be presented centrally, use stimPos=0 0
- if you want a stimulus to be presented next to the previous stimulus, omit defining its position, eg. stimPos=-100 0;;200 0 will present stim1 100px left from a central position, stim2 next to it, and stim3 200px right from a central position
- if you define stimPos for fewer stimuli than you have on the screen, the last stimulus for which there is a position defined and the stimuli for which there is no position defined will be positioned as a group; eg. stimPos=-100 -100;0 0 for six stimuli will present stim1 100px to the left and 100px up from the center, and stim2 to stim6 as a block of five stimuli centered horizontally and vertically
- you can use this option to shift an entire block of image, eg. stimPos=100 100 with six stimuli will effectively move all stimuli
- you can define only the horizontal displacement, eg. stimPos=100 is equivalent to stimPos=100 0
- you can also use the following labels: left, right, top, bottom, center. Eg. stimPos = left top will position the stimulus to the left and the top of the screen.
- does not work for target!

mask {*filename*}
- A mask can be used to reduce or illuminate afterimages. If a mask is used, enter the mask filename. The mask must have the same image format (e.g. jpg) as the stimuli

- In sequential presentations, the mask will appear during ISI, between all screens (but not after last screen in case last screen has limited presentation time). For simultaneous presentations (trials with only one screen), the mask will appear for unlimited time after the screen

trialText {*text*}

- can use this to display text above the stimuli during the trial
- if multiple texts are needed during one trial, use {} to delimit each text, eg. trialText={text1}{text2}{text3}
- text will change from screen to screen according to the options selected in trialTextOptions

trialTextOptions {*ITI;ISI;firstStim;lastStim;afterStim;allTrial;allStim;withStim;withResponse*}

- used to specify when trialText appears on screen
- ITI: trialText appears during ITI
- ISI: trialText appears during all ISI
- firstStim: trialText appears while first stimulus is presented
- lastStim: trialText appears while last stimulus is presented
- afterStim: trialText appears after all stimuli are presented (last stimulus must have limited presentation time)
- allTrial: trialText appears during the whole trial (immediately after ITI)
- allStim: trialText appears during the whole stimuli presentations, including ISI (if last stimulus has limited presentation time, trialText will disappear when last stimulus disappears)
- withStim: trialText appears during stimuli presentations only (excluding ISI)
- withResponse: trialText appears during the window when responses are enabled. This option overrides all other
- default: trialText = lastStim;afterStim (the text will appear with the last stimulus and remain on the screen until participant responds and experiment advances to the next trial)

trialTextPos {*above,below,center,number_in_px*}

- The text will remain on the screen until the experiment advances to the next trial (e.g. if the stimuli are presented for a limited time, the text will still be on the screen after the stimuli are hidden)

fixation {*ITI;ISI;allTrial;allStim;withStim*}

- select when to display fixation cross (leave blank if fixation is not required for a trial)
- can use one or more options (separated by semicolon)
- anomalies (e.g. fixation=ISI with simultaneous presentations) are ignored

Examples

1) fixation = ITI

- fixation cross will be displayed before the start of the trial (during ITI)

2) fixation = withStim

- fixation cross will appear on top of the stimuli only

3) fixation = allTrial

- fixation cross will appear throughout the trial, on all screens (i.e. between and after stimuli as well)

4) fixation = allStim

- fixation cross will appear throughout the trial, on all screens until presentation of stimuli is finished (i.e. if last stimuli have a limited presentation time, fixation cross will disappear when this presentation time is up)

5) fixation = ISI

- fixation cross will appear on screens between stimuli during a sequential presentation

Legacy

- if fixation = 1, fixation cross will be displayed during ITI (equivalent to fixation = ITI)
- if fixation = 1 and fixationStim = 1, fixation cross will be displayed during ITI and with the stimuli (equivalent to fixation = ITI;withStim)

ITI {*milliseconds*}
- inter-trial interval in milliseconds

ISI {*milliseconds*}
- inter-stimuli interval(s) in milliseconds
- used to define sequential presentations of stimuli
- you can have multiple numbers, separated by semicolons; each number is one ISI (i.e. the time between two stimuli)
- e.g. ISI = 1000 for a trial with two stimuli -> there will be a 1000ms interval between stim1 and stim2
- if you want two (or more) stimuli to appear on the same screen, simply omit defining an ISI between them
- e.g. ISI = 1000 for a trial with three stimuli -> there will be a 1000ms interval between stim1 and the following screen presenting stim2 and stim3 simultaneously
- e.g. ISI = 1000;;2000 for a trial with four stimuli -> stim1, followed by an interval of 1000ms, then stim2 and stim3 simultaneously, followed by an interval of 2000ms, then stim4
- note that not defining an ISI is different from setting that ISI to 0
- e.g. ISI = 1000;0;2000 for a trial with four stimuli -> stim1, followed by an interval of 1000ms, then stim2, followed immediately (ISI=0) by stim3, followed by an interval of 2000ms, then stim4
- in general: number of screens with stimuli = number of non-blank elements in ISI + 1
- if you need the same ISI multiple times (between all screens), you can use format milliseconds*times, eg. ISI = 1000*5 is equivalent to ISI = 1000;1000;1000;1000;1000

presTime {*milliseconds*}
- presentation time in milliseconds for test trials containing images or words, and for instructions trials. If not specified, stimuli or instructions will stay on the screen until the participant responds
- if you need the same presentation time for all screens, you can use format milliseconds*times, eg. presTime = 1000*6 is equivalent to presTime = 1000;1000;1000;1000;1000;1000

Legacy

presTime2 {*milliseconds*}
- presentation time in milliseconds for the second screen with stimuli in a sequential presentation. If not specified, the second screen with stimuli will stay on the screen until the participant responds

key {*any number, character, text*}
- specify the correct response for a trial; the field is case-insensitive
- if participants respond by clicking buttons, the correct key is the number of the button (e.g. 2) rather than the name of the button (e.g. button2) or the text displayed in the button (e.g. happy)
- if participants respond by clicking on the stimuli, the correct key is the number of the stimulus (e.g. 2) rather than the name of the stimulus column (e.g. stim2) or the stimulus name (e.g. happyFace)

feedbackCorrect {*text*}
- the text specified here will be displayed after each correct response from the participant

feedbackIncorrect {*text*}
- the text specified here will be displayed after each incorrect response from the participant

feedbackTime {*milliseconds*}

- presentation time for feedback When time is up, Testable will advance to the next trial. If not specified, feedback will stay on the screen until the participant clicks the NEXT button

button1 {*text*}
button2 {*text*}
button3 {*text*}
button4 {*text*}
button5 {*text*}
button6 {*text*}

- create buttons with customised text. Enter the desired text in these columns. The buttons can be used for advancing to next trials (e.g. for instructions), or to record responses from participants (e.g. “Next”, “Continue”, “Start”, “Let’s practice”, “Same”, “Different”)

keyboard {*0 1 2 … 7 8 9 a b c … x y z space enter left right up down*}

- enables the keys specified here. E.g. “1 2” will enable the keys 1 and 2; “a b” will enable the keys a and b (the field is case-insensitive). All other keys will be unresponsive. Used to record responses from participants and/or advance to next trials.

responseWindow {*timeout OR start_timeout OR firstStim;lastStim;afterStim; withStim*}

- used to define the time during which participants can respond
- **start** can be defined with labels (***firstStim***, ***lastStim***, ***afterStim***) or numbers (***milliseconds*** from the beginning of the trial)
- **timeout** is specified with numbers (***milliseconds*** after responses are enabled)
- ***withStim*** enables responses only during presentation of stimuli
- start is optional (default is ***lastStim****)*
- column waitStimEnd=1 will override everyhting and is equivalent to responseWindow=afterStim;
- when time is up, Testable will advance to the next trial and the response for the trial will be marked as 99 (i.e. timed out).
- by default (i.e. when responseWindow is not specified), participants have unlimited time to respond starting with the last stimulus displayed

Examples

1) responseWindow not defined
- responses are enabled when last stimulus is displayed, for unlimited time

2) responseWindow = 2000
- responses are enabled when last stimulus is displayed, for 2000ms

3) responseWindow = 0_2000
- responses are enabled from the start of the trial for 2000ms

4) responseWindow = 1000_2000
- responses are enabled 1000ms after start of trial, for 2000ms

5) responseWindow = 1000_
- responses are enabled 1000ms after start of trial, for unlimited time

6) responseWindow = withStim
- responses are enabled only while stimuli are displayed

7) responseWindow = lastStim
- responses are enabled from when last stimulus is displayed for unlimited time

8) responseWindow = lastStim_2000
- responses are enabled from when last stimulus is displayed for 2000ms

9) responseWindow = afterStim
- responses are enabled after last stimulus finished displaying for unlimited time

10) responseWindow = afterStim_2000
- responses are enabled after last stimulus finished displaying for 2000ms

counterBlock {*na,1*}

- displays “Block x of y” in the top left corner of the screen only during instructions trials. To be used for multiple blocks/experiments. A block is considered an uninterrupted series of “test” trials.

counterTrial {*na,1*}

- if “1”, displays “Trial x of y” in the top left corner of the screen during test trials

timer {*na,1*}

- if “1”, shows the remaining presentation time (from presTime) in the top right corner during test trials

select {*na,1*}

- select the trials you want to include when computing an average correct score for the experiment (otherwise the average correct score is computed for all trials)

bgColor {*HEX color code* OR *red, green, blue, etc.*}

- overrides the default background color from the Setup. Popular colors can be defined by name (e.g. “black”, “red”, “green”).

textColor {*HEX color code* OR *red, green, blue, etc.*}

- overrides the default text color from the Setup. Popular colors can be defined by name (e.g. “black”, “red”, “green”).

random {*na, 1, 2, …*}

- used to mark the trials that should be randomised. Any continuous group of trials marked with the same number will be randomised within.

randomBlock {*na, 1, 2, …*}

- used for randomisation of blocks. Mark each block to be included in the randomisation with the same number (don’t forget to include the instructions if necessary). The order of blocks will then be randomized
- also used for multiple level block randomisation. Column randomBlock can now have multiple numbers (separated by a space) to specify multiple levels for block randomization. The first number is the first-level block, the second number is the second-level block, and so on. Let's say you have ten trials with the following information under randomBlock:

  1 1
  1 1
  1 2
  1 2
  1 3
  1 3
  2 1
  2 1
  2 2
  2 2

- In this case, first six trials form the first-level block 1, while last four trials form the first-level block 2. Within first-level block 1, there are three second-level blocks, each with two trials. Within first-level block 2, there are two second-level blocks, each with two trials. Within first-level block 1, the three second-level blocks will be presented in a random order. Within first-level block 2, the two second-level blocks will be presented in a random order. Finally, the presentation order of first-level blocks 1 and 2 will be randomized. If you mark a block with 0, it will not be included in the block randomization.

randomPick {*number*}

- used for random selection of a subset of trials. Every time an experiment's link is accessed (i.e. for every participant completing the experiment), the algorithm will randomly select for presentation N trials from the defined set of M trials, where M is the number of consecutive trials marked with number N under randomPick. For example, if you want to present a subset of 20 trials from a larger set of 100 consecutive trials, each of the 100 trials needs to have 20 under randomPick.

subjectGroup {*na, 0, 1, 2, …*}

- used to run experiments between groups. Mark with 0 the trials you want shown to all subjects (e.g. initial instructions), with 1 the trials to be shown to group 1, with 2 the trials to be shown to group 2, etc. If the between subjects functionality is not needed, omit the column or have “na” in all fields.

SUGGESTED COLUMNS FOR ORGANISATION OF TRIALS

- these are not used by the platform, but may help you organise/read the results

test {*text*}
subTest {*text*}
condition1 {*text*}
condition2 {*text*}
condition3 {*text*}
condition4 {*text*}
trialNo {*number*}

- can be helpful when you randomise trials to easily identify when each trial was presented

How to create forms (surveys)

Forms with multiple question and response types can be easily created using the same spreadsheet and many of the columns you normally use for stimuli presentation. To create forms, you need to mark the trials as “form” under column ***type***. The trials are basically the questions that create a form. Here are the columns you can use for forms.

type {*“form”*}
- to create a form, you need to have type=form

head {*text*}
- used to define the question

body {*text*}
- used to provide additional details on the question, if necessary. The text here appears in smaller font than the text under **head**

responseType {*box,comment,radio,checkboxes,dropdown,slider,stars*}
- this is where you specified how participants should respond (you can also leave it blank, in which case there won’t be any response options for participants - you can use this option to provide more information about the survey)
- soon: ***rank***

responseOptions {*text*}
- use this to define options for the selected response type
- radio/checkboxes/dropdown: specify options separated by semicolon (e.g. poor;average;good)
- slider: use format
  - min_max_step;start;flags{hideStart,hideMinMax,hideGrid;hideValue}
  - e.g. 0_100_5;50
  - e.g. 0_100_5;50;hideGrid;hideStart
  - works also for labels, in this case you need to define all steps;
  - e.g. poor_average_good;average

responseOther {*text*}
- gives participants the possibility to select an answer that is not predefined

responseRows {*text;text;…*}
- for cases when you want participants to provide answers on multiple dimensions for the same questions (e.g. rate biscuits on multiple attributes)
- specify all dimensions separated by semicolon

required {na,*1*}
- mark if an answer is required

separator {na,*1*}
- add a visual separator (dotted line) from previous question

pageBreak {na,1}
- use to split questions on multiple pages. By default, there will be a “NEXT” button added at the bottom of the page. If you want to change the text on this button, use column button1

pageName {text}
- form title (can be different on every page of the survey)

separator {na, *1*}
- use to add a visual separator (dotted line) from previous question

IMPORTANT!!
Many columns/parameters used for stimuli presentations can also be used here to similar effects. Some examples:
- randomisation within and between-blocks (see columns **random** and **randomBlock**)
- random selection of a subset of trials (see column **randomPick**)

You can also add images to each trial in the forms by using columns **stim1…stim6** and **stimFormat**, just like for stimuli presentation.

How to get the results

Results will be saved in one .csv file per participant. The results file will basically be the uploaded spreadsheet with information about the experiment, with four additional columns:

timestamps {*milliseconds*}
- records the timestamp of each response provided by the participant. It can be useful to look at how long the participant spent on an experiment, for example

responses {*key/number*}
- records participant's responses

RTs {*milliseconds*}
- records response times

correct {*0,1*}
- records whether response was correct or incorrect. It gives 1 if response matches the information in column **key**, and 0 otherwise.

Depending on the options you selected for your experiment (e.g. if you chose to randomize your trials), two additional columns may be saved:

orderFile {*number*}
- records the original order of trials in your trial file

orderPres {*number*}
- records the order in which trials were presented during the experiment (consecutive numbers)

RTkeys {*milliseconds;milliseconds;…*}
- records response times for each key pressed in trials required free text input from participants

The results files can be viewed and downloaded when clicking on the "Results" button in the main projects dashboard.

Compatibility issues

In general, the experiments look the same regardless of the browser used to access them. There are only two issues you should be aware of:

- **keyboard responses do not work in Firefox**. Whenever an experiment requires keyboard responses and it is run in Firefox, a warning appears **THIS HAS NOW BEEN SOLVED!!**
- in **Safari**, if the experimenter selects to save the results locally, the results file gets loaded in a new tab in the browser (instead of being downloaded). You can then save it.

Otherwise the browser should not matter. However, I recommend Chrome whenever possible.

Testable currently does not support data collection via mobile devices.