

Collector Documentation

Documentation by Anthony Haffey on behalf of [Some Open Solutions](#).

If you identify bugs or errors please try to save and refresh the page. Please do report any bugs or errors at:

<https://collectalk.com/categories/bugs-and-errors>

If you have suggested clarifications, please go to
<https://collectalk.com/categories/documentation>

The most up to date version of this documentation can be found at:

https://docs.google.com/document/d/1SKYIJF1dAjMDS6EHUIwfZm2KQVOzx17S6LbU_oSGxdE/edit?usp=sharing

License

Collector (Garcia, Kornell, Kerr, Blake & Haffey)

A program for running experiments on the web

Copyright 2012-2016 Mikey Garcia & Nate Kornell

This program is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License version 3 as published by the Free Software Foundation.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program. If not, see <<http://www.gnu.org/licenses/>>

Kitten release (2019-20) author: Dr. Anthony Haffey (a.haffey@reading.ac.uk)

Table of Contents

Collector Documentation	1
License	1
Table of Contents	2
Using Collector online (i.e. without installing Collector)	4
Dropbox and google registration	4
Installing Collector on your machine	8
Advantages	8
Installing with Python	9
Creating experiments	12
Conditions sheet	13
Trial type editor	13
Graphic editor	13
How it looks	16
Timeline	18
Capturing response with keyboard	20
Capturing the response with the mouse	21
Changing the canvas size	21
Code editor	22
Saving responses	22
Default javascript and css packages	22
Collector functions	23
Procedure sheet	24
Shuffle 1, 2, 3 etc.	24
shuffle 2, 3, 4 etc.	25
Trial type	26
Item	27
Max time	27
Text	27
Weight/freq/frequency/repeat - useful for piloting your task	27
Stimuli sheet	27
List Stimuli button	28
Cell editor	29
Helper sidebar	32
Downloading and Uploading experiments	32
Running your study	33
Save!	33
Both Online and Installed versions	33
Click the “run” button	33
Publishing your task	34

Saving for installed version only	36
Have multiple studies in a row	36
Surveys	38
Type	39
answers	40
values	40
score:[your score name]	40
feedback	40
feedback_color	40
item_name	41
no_text	41
optional	41
shuffle	41
text	41
Data	43
Saving data with google drive	43
Saving and emailing data with a server	49
Telling Collector where your saving/emailing script is	49
Decrypting your data (and understanding it)	49
Decrypting your data within the “Data” tab	51
Understanding your data	52
FAQ/General questions	53
Can I use Collector if I am not a programmer?	53
What language is Collector written in?	53
Do I need to ask anyone if I want to use Collector?	53
Troubleshooting	53
Further notes	54
What happens with my password for encryption and decryption	54
What happens with my password for Collector	54
Contributing	55
Original contributors	55
Tyson Kerr	55
Victor Sungkhasettee	55
Adam Blake	55
Nate Kornell	56

Using Collector online (i.e. without installing Collector)

To use without installation go to one of the following pages:

- <https://some-open-solutions.github.io/ocollector>
(allows you to choose which release of Kitten you want)
- <https://ocollector.org>
(allows you to choose which release of Kitten you want)
- <https://some-open-solutions.github.io/collector>
(will use the most recent version of kitten that is stable - not yet what we'd recommend)

To use without installing, you will need to register with Dropbox and Collector

Dropbox and google registration

Whilst this can take half an hour, there are two very strong advantages of registering a new account with google, and then using it to create a dropbox account

- You will be able to share access to the dropbox account with collaborators without jeopardizing any of your personal files on dropbox.
- You will be able to use the google account for storing up to 50,000 files a day for free!

The easiest way to do this is to register a google account first.

Once you've registered a google account, go to dropbox and click on the **create an account link**:



Sign in

or [create an account](#)



Sign in with Google



Sign in with Apple

or

Email

Password

This page is protected by reCAPTCHA and is subject to the [Google Privacy Policy](#) and [Terms of Service](#).

☒ Remember me

Sign in

[Forgotten your password?](#)

And then the **sign up with google** button:



Create an account

or [log in](#)

First name

Surname

Email

Password

This page is protected by reCAPTCHA and is subject to the [Google Privacy Policy](#) and [Terms of Service](#).

☐ I agree to the [Dropbox Terms](#).

Create an account

or



Sign up with Google

Now you can log in to Collector with this new account. Go to either:

<https://some-open-solutions.github.io/ocollector>

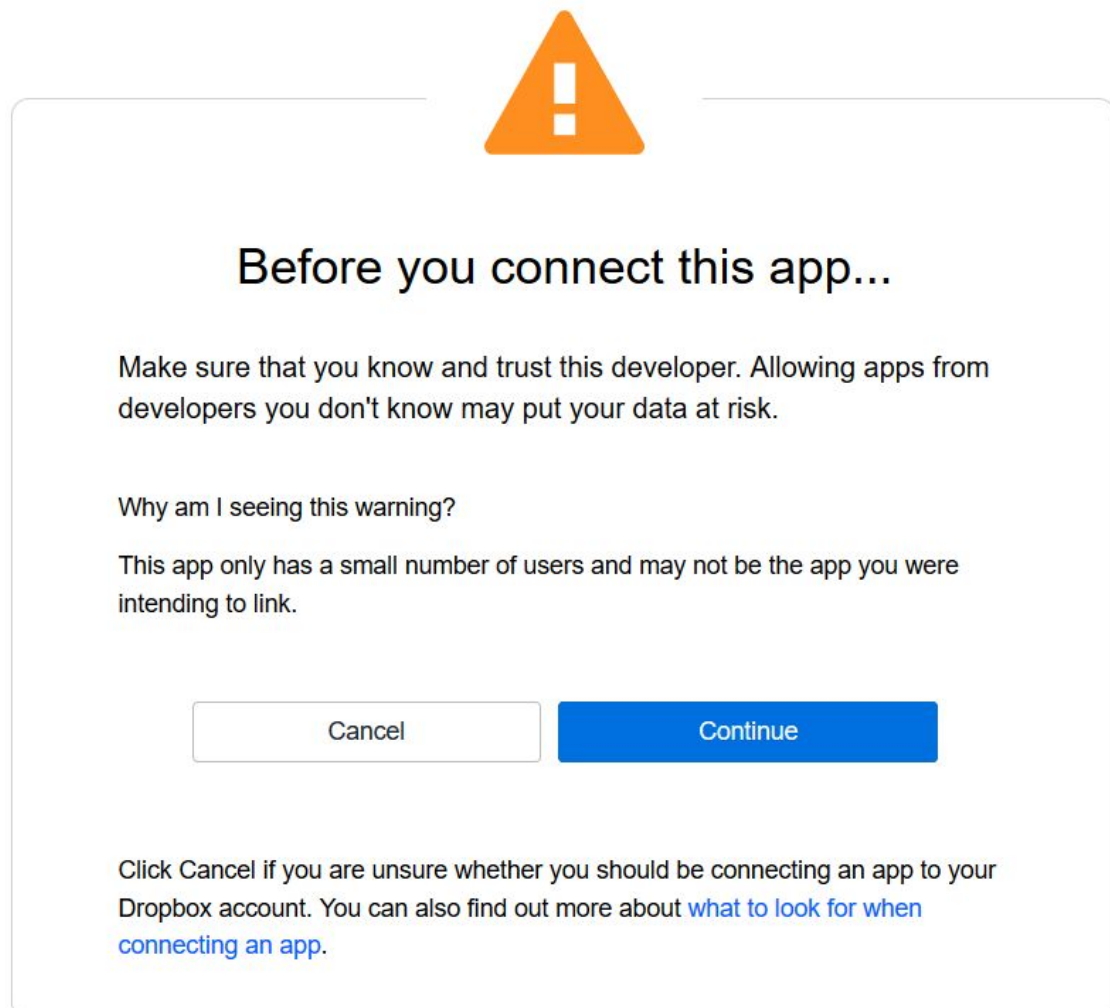
Or

<https://ocollector.org/>

And choose the version you want:

And then open “kitten” (as this is still developmental software, you’ll be able to use “Cat” when we’ve finished the development phase).

When you’re asked to login with dropbox you can use the account you just created! When you click through to dropbox, you should see this:



This message is because it’s possible to create dropbox apps that edit your main dropbox files. Whilst this Collector app should only edit files in it’s own “Collector-SOS” folder, it is generally not a good idea to use a personal dropbox account with a relatively new app that gives you this sort of warning.

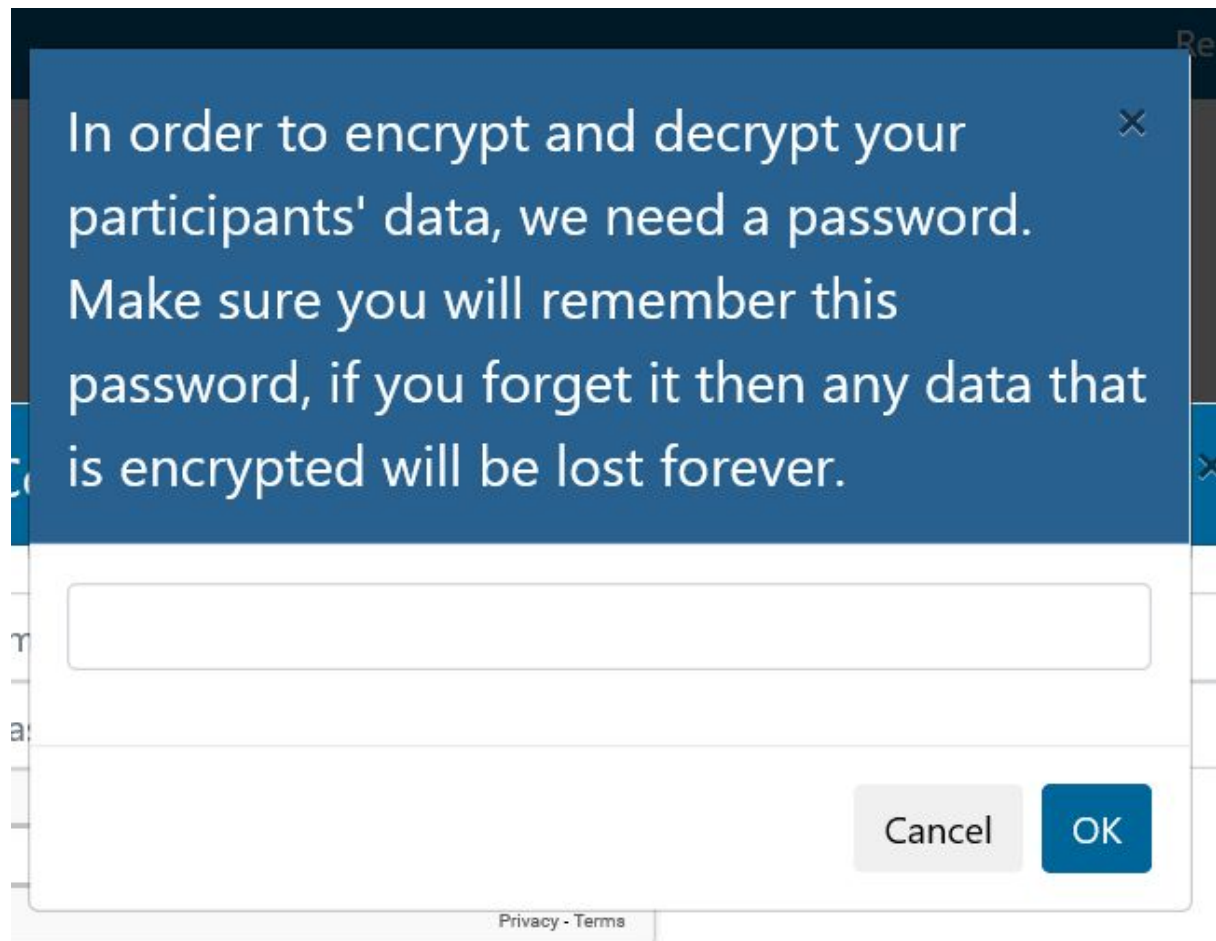
So if you haven’t yet created a new gmail account for your research and linked it with dropbox, please do this now.

If you are using a new gmail/dropbox account you created specifically for this, then click “Continue”.

You should see the following, in which you can click “Allow”:



This should take you back to Collector. Collector will then set up your dropbox account and ask for the password you want to use for data encryption and decryption:



Think carefully about what your password will be. **Do not use a password you use for anything else!** Here are some reasons why:

- You may want to share this dropbox account with a collaborator
- Whenever possible, you should not use the same password on different websites.

However, having said that, your password will never be stored (see [What happens with my password for encryption and decryption](#) for clarifications of how your password is used to decrypt data).

You can skip the next section as this is for installing Collector, which you don't need to do now.

Installing Collector on your machine

Advantages

There are some advantages of installing Collector on your machine and then pushing the changes to a github repository where your site can be hosted (all for free):

- You don't need to wait for files to be saved on Dropbox before seeing changes to your task. This can save time with previewing.
- Your task might run a little smoother if you are using images, videos or audio files. This is because you can upload these media files onto your github repository, so there is no cross-site delay between github and dropbox when requesting for the images, videos etc. However, hopefully the **buffering** that Collector does should avoid or minimise any delay caused by calling in files from dropbox into your experiment.

Installing with Python

Not recommended unless you're quite confident with Python and willing to learn how to use github!

Hopefully in the near future there'll be stable ways of installing Collector without needing Python on your machine. However, in the mean time you will need Python 3 to install Collector.

The first step is to download Collector is to download the installer at:

<https://raw.githubusercontent.com/some-open-solutions/collector/master/Installer/installer.py>

You should be able to right click and select "save link as".


You will probably want to run this installer using the command line so that you can receive messages about any packages you need to install. You will need to the following packages before the installer and Collector will work:

- eel
- github
- io
- json
- os (I think this is a default package)
- platform
- pygit2
- shutil (I think this is a default package)
- tkinter (I think this is a default package)

Once you have Python 3 with the packages above, the installer should do all the work for you. Once Collector is installed, you should be able open it in the command line (once you've navigated to the directory) with either:

"python Collector.py"
or

“python3 Collector.py”

Now, you need to create an empty repository on github to synch up your repository with. Hopefully you'll find registering with github intuitive enough. Once you've registered, you can create a **“New repository”** by clicking on the  button in the top right corner. There are some important settings in the following page:

1


Owner


Repository name *

Great repository names are short and memorable. Need inspiration? How about [bug-free-train?](#)

Description (optional)

2

☒  **Public**
Anyone can see this repository. You choose who can commit.

☐  **Private**
You choose who can see and commit to this repository.

3

Skip this step if you're importing an existing repository.

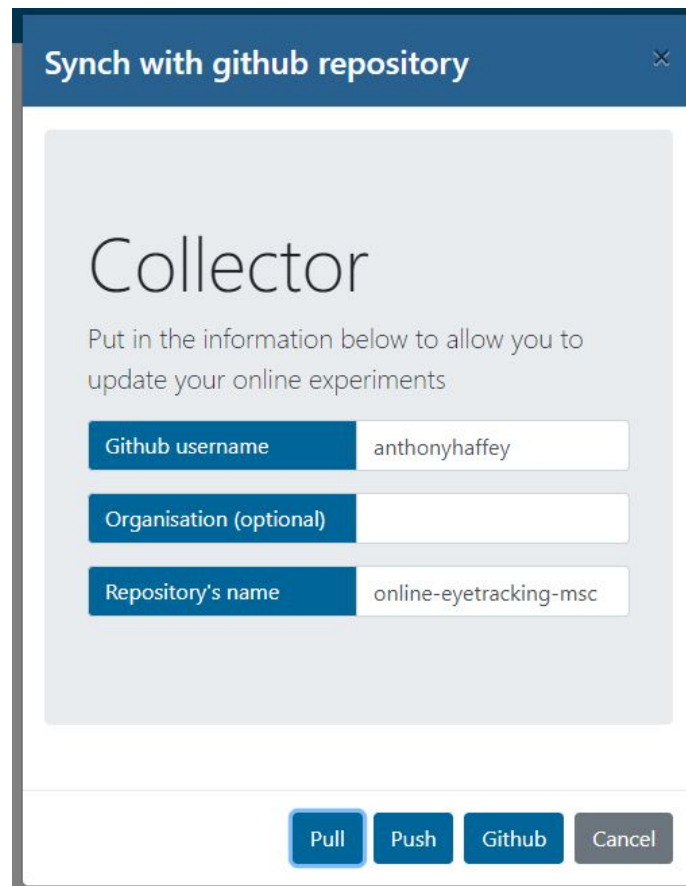
☒ **Initialize this repository with a README**

This will let you immediately clone the repository to your computer.

1. If you plan to collaborate with anyone, you may want to change the **owner** to an **organization**. This will make it easier for multiple people to **push** changes to the website (i.e. git repository). If you haven't yet created the **organization** you need, you should do this before proceeding.
2. Make sure to leave this on “public”. Otherwise participants won't be able to see your website! (although technically you can make it public later)
3. Do not tick this box! You want your repository to be completely blank so that when you do push changes to it there'll be nothing to block you.

Once you have created your repository, you will be ready to synch your local installation of Collector with it. Open Collector.py in the command line as described above with “python Collector.py” or “python3 Collector.py”. Once you're in you'll be talked through a series of steps to get you started, such as coming up with a password. Importantly, **this password does not have to be, and should not be the same as your github password**. This is because you may want to share your repository at some point in the future, and this will involve sharing this password to allow collaborators to decrypt the data.

Once you're through those steps click on the **github icon** in the top right corner  and you should see something like:



Synch with github repository

Collector

Put in the information below to allow you to update your online experiments

Github username: anthonyhaffey

Organisation (optional):

Repository's name: online-eyetracking-msc

Pull Push Github Cancel

Use the same github **username** you did to register.

Put in the **organization** name if you created the repository in an organization. If not, you can skip this.

Put in the Repository name.

Before proceeding, there are some key concepts to clarify:

- **Pull** is when you update your local installation by pulling in changes from the online repository (i.e. your website). This would make sense when you have a collaborator who just updated the repository.
- **Push** is how you update your website/repository.
- The **Github** button will just open your github repository on a new page.

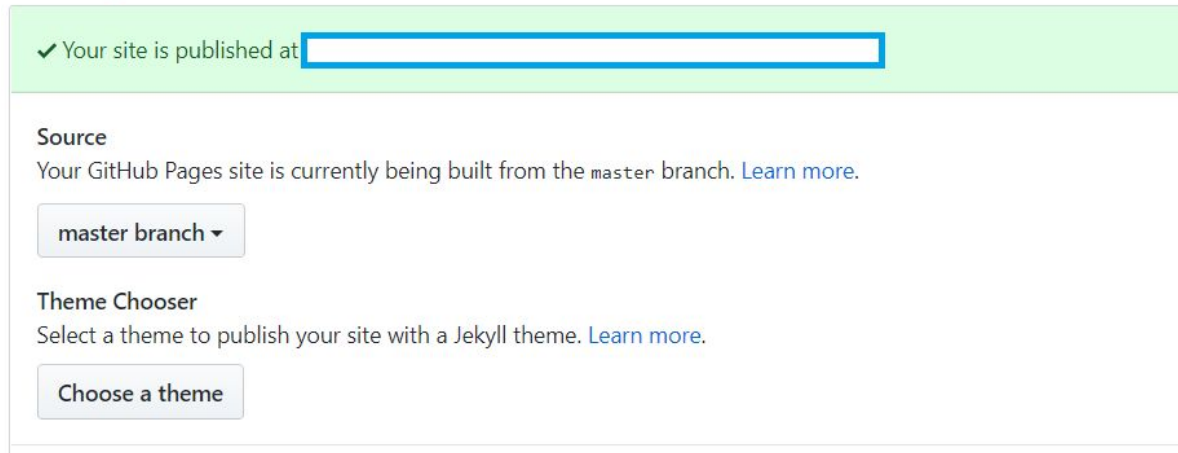
In our case, we want to **push** our local installation of Collector into the repository, so now click on “**push**”. You’ll be asked for your password you gave when registering with github. This will happen every time you push changes because Collector won’t store your password for updating github. (It doesn’t really store your password for decryption either, see [What happens with my password for encryption and decryption](#)).

Assuming all went well, your github repository will now have your installation of Collector, and you’ll be very nearly ready to start creating experiments.

The final thing you need to do is tell github to treat your repository as a website. To do this, go to the repository in github and click on **Settings**. Scroll down until you get to Github Pages:

GitHub Pages

GitHub Pages is designed to host your personal, organization, or project pages from a GitHub repository.



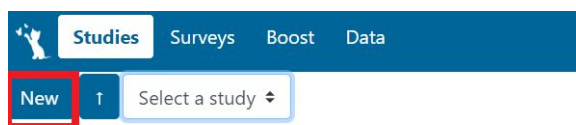
Change the Source from “None” to “master branch”.

After a few minutes, your website will be live! You should be given a url for your website where there's a white box with a blue outline above, but it may take a few minutes before it works.

Creating experiments

For the purposes of this documentation, I'll be using the example of creating a stroop task called “stroop_example”.

To create a new experiment just click on the “New” button at the top left of the screen:



Conditions

Procedure



Stimuli



If you're developing your experiment online rather than with an installation, give it a few seconds for the experiment creation to finish. You'll know it's ready because the new name will be where "Select a study" was before:



Conditions sheet

(Can be edited directly as a spreadsheet, e.g. using Excel, in the installed version of Collector - go to the User folder)

The conditions sheet controls some of the settings of the experiment. For example, if you want the user to have a randomised id rather than what they typed, then change the cell just below "participant_id" from "on" to "random":

Conditions

1	Name	Notes	Stimuli	Procedure	fullscreen	welcome	participant_id	end_message	start_message	buffer
2	condition_1	You can put me	stimuli_1.csv	procedure_1.cs	off		on			5
3										



1	Name	Notes	Stimuli	Procedure	fullscreen	welcome	participant_id	end_message	start_message	buffer
2	condition_1	You can put me	stimuli_1.csv	procedure_1.cs	off		random			5
3										

Have a look around at the different settings (notice how the "Help" sidebar tells you about each of them as you click across different columns).

If you want to have different participants experience different conditions, then describe the differences in the different rows. This will result in you creating slightly different links for each version of the task.

The differences between the different conditions will be due to differences between the different **procedure** and **stimuli** sheets. To create a new procedure or stimuli sheet, click on the "New Sheet" buttons in their respective sections.

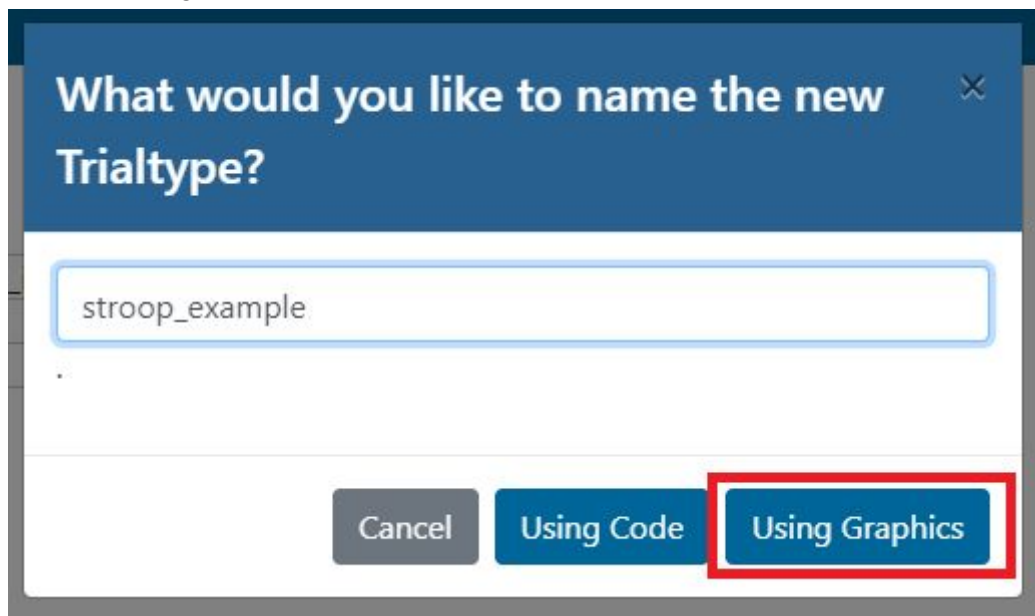
Trial type editor

Graphic editor

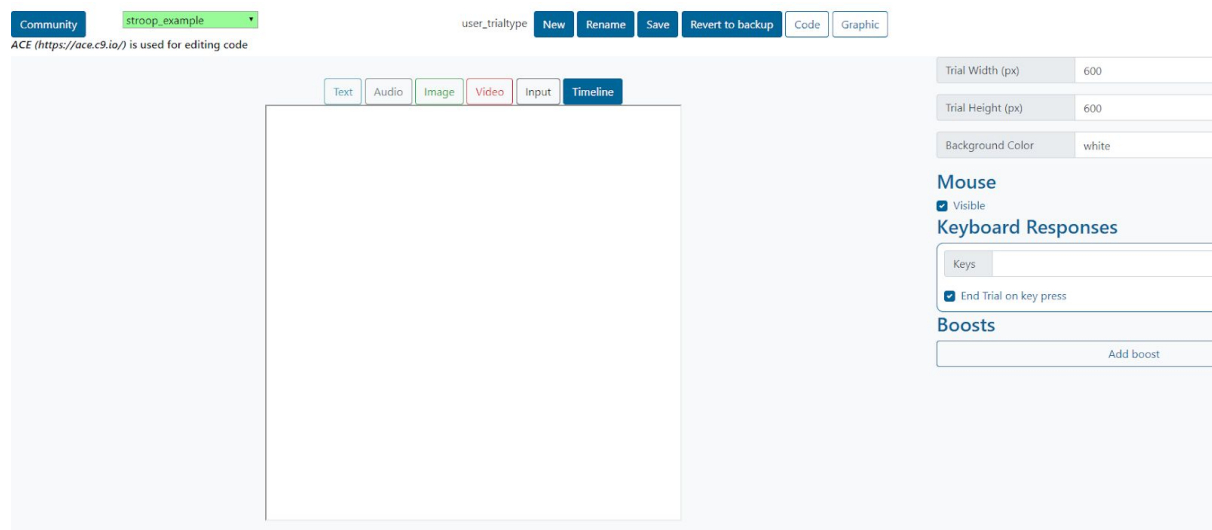
Trial types are types of trials the participant will experience. To create a new trialtype just click on the “New” button towards the bottom of the page:



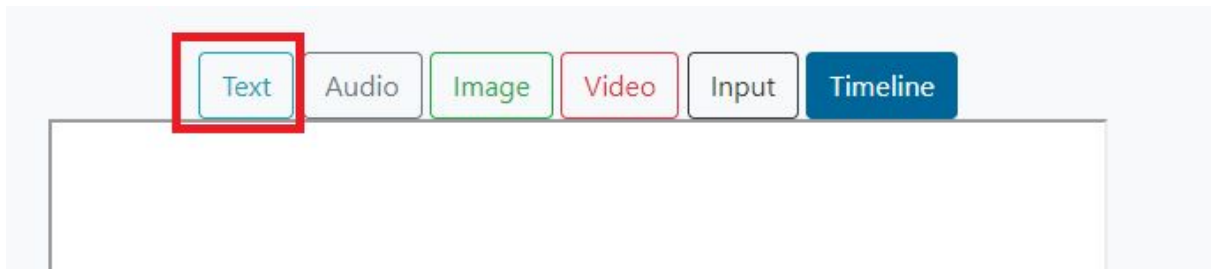
And then you'll be asked for a name and whether you want to create the trialtype using code or the graphic editor. Let's start by talking through how to use the graphic editor. So let's click on “Using Graphics”:



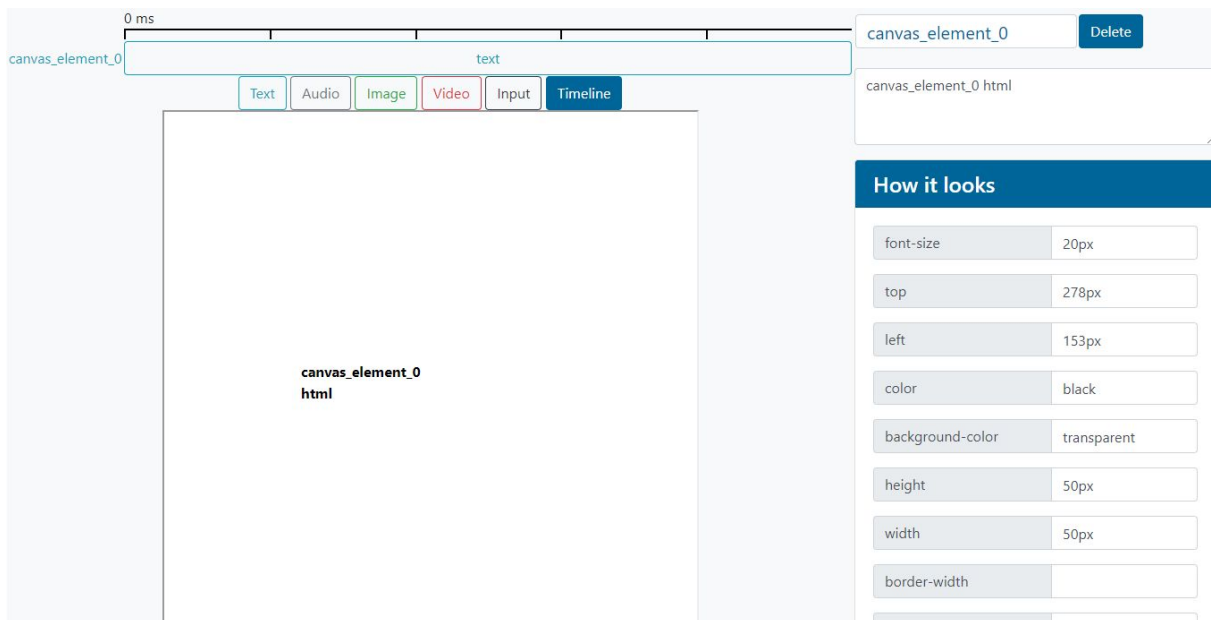
You may have noticed that I've given exactly the same name to the experiment and the trialtype “stroop_example”. I find this helpful as they can never get confused with each other. Once you've created the new trialtype you should see the the graphic editor:



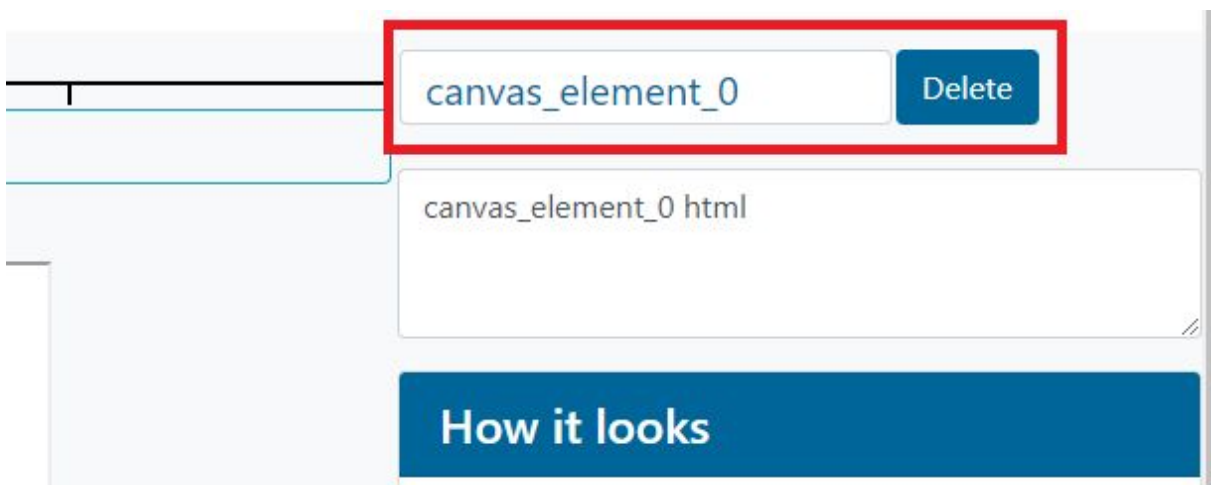
For a simple stroop experiment, we'll start by adding some text. Just click on the **Text** button:



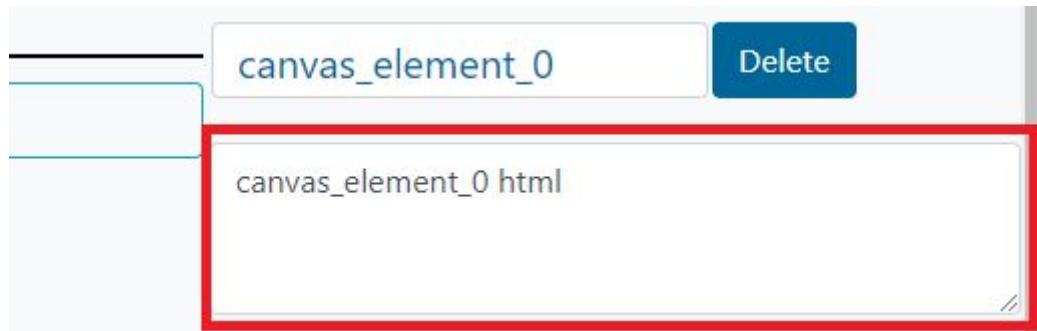
And then click where you want it. I'm going to try to put this pretty close to the center of the page:



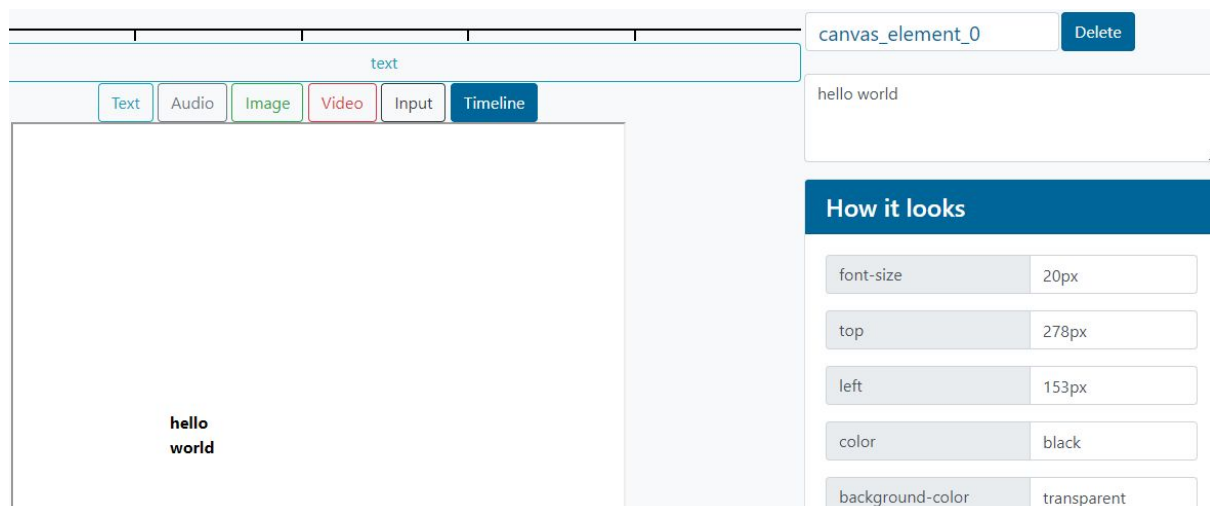
Let's unpack what we now have. In the top right corner we have the element_id "canvas_element_0":



This can be useful for making the experiment interactive, but we don't need to worry about that for now. Just below that we have the actual text in this element:



As it stands, every time you use the trial “stroop_example” you will get the text “canvas_element_0 html” - which is not particularly helpful. Let's change that to just say “hello world” to see exactly how this works:

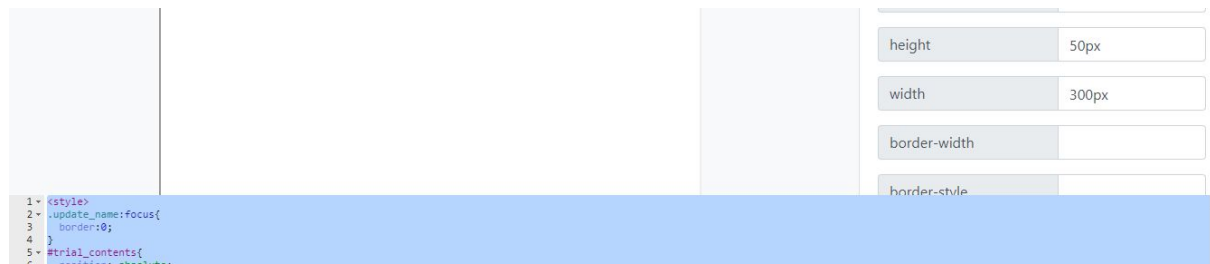


So you could theoretically create a separate trial with a different word or series of words for each trial. However, that would be very time consuming, so we're going to embed a **variable** instead. **Variables** change for each trial depending on what you have in your procedure and stimuli spreadsheets. You just write the column title in the `{}` brackets. For a stroop task, we'll make use of the fact that **stimuli** sheets already have a **cue** column. So let's just refer to that column (we'll edit the **stimuli** sheet later). We'll replace “hello world” with “`{{cue}}`”.

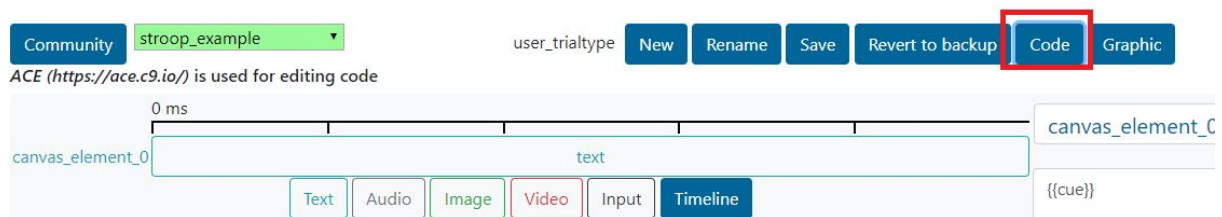
How it looks

Play around with the “How it Looks” panel to see the different sizes and colors you can have.

You might find that the **code** editor is in the way of the lower options:

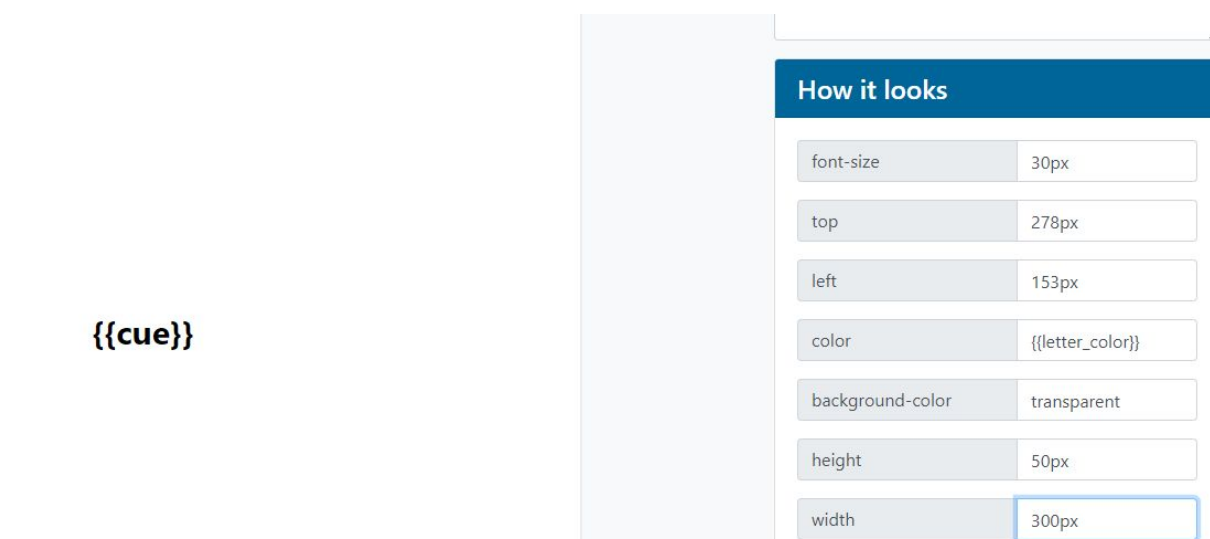


You can hide the **code** editor by clicking on the “**Code**” button at the above the **graphic** editor:



You might need to click it a couple of times before the **code** editor disappears. You can click it again to make it come back. This can be a nice way to learn code - as you edit the experiment using the **graphic** editor, the code is written for the **trial type**. But let’s get back to making our stroop task.

Being a stroop task, **color** is something that will also have to be a variable that changes from trial to trial. So I’m going to replace “black” with “`{{letter_color}}`” which is a column I’m going to need in my stimuli sheet later. I’m also going to make the font-size 30px; and the width 300px:



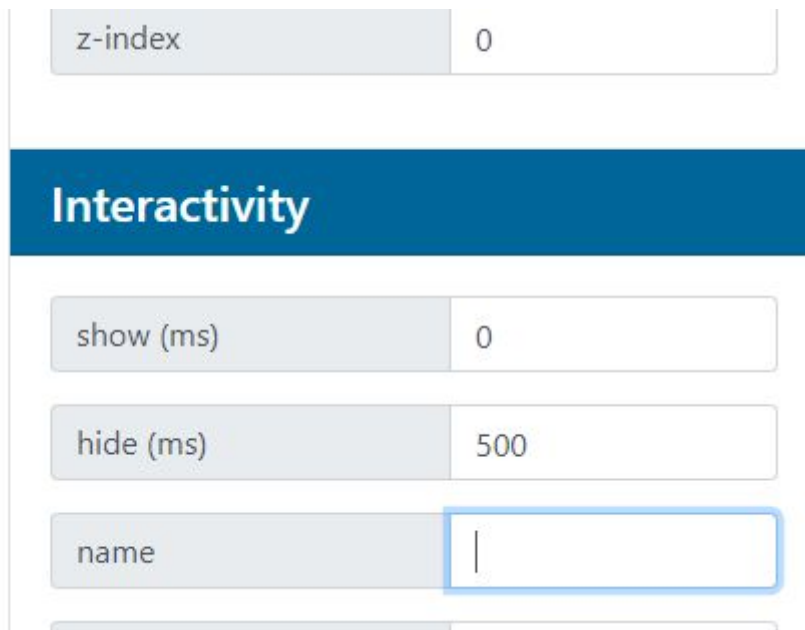
It’s important to have an appropriate width; if your width is too small your text will run over multiple lines. You might have noticed this in the earlier example with “hello world”.

Timeline

At the top of the graphic editor, we get a sense of what happens when. In our case, let's:

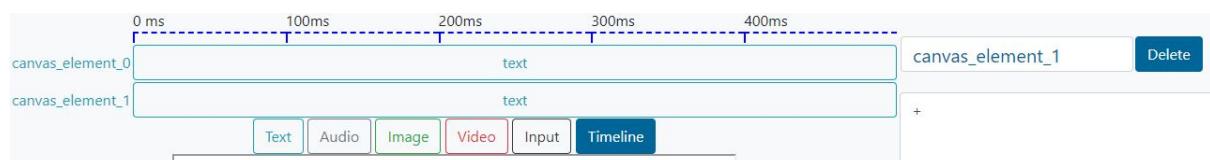
- Add a fixation cross at the start for 500ms
- Add a 500ms gap following this by changing the start time of the word to 1000ms

To add a fixation cross, we just add another text element as described above. I'm going to make the **font-size** 100px and put it 250px from the top and left. This should put it in the center of a 600 x 600 pixel trial. To make it only be present for 500ms I'm going to scroll down to the **interactivity** panel, and change the **hide(ms)** value to 500ms:



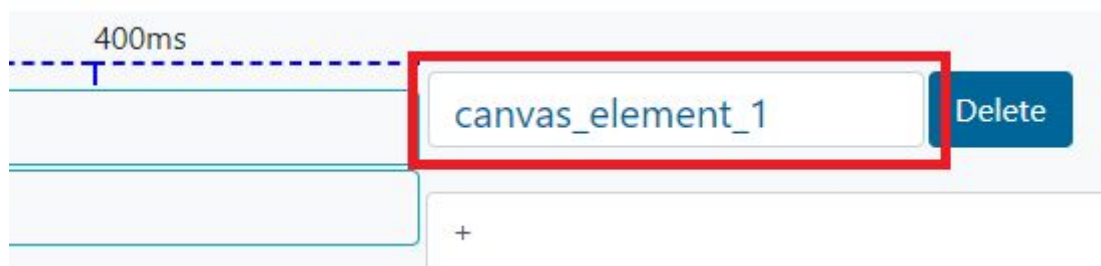
(Note that you can click on **how it looks** or **interactivity** headers to hide and show these interfaces).

Now that we've changed the **hide** time for the fixation, our timeline at the top looks like this:

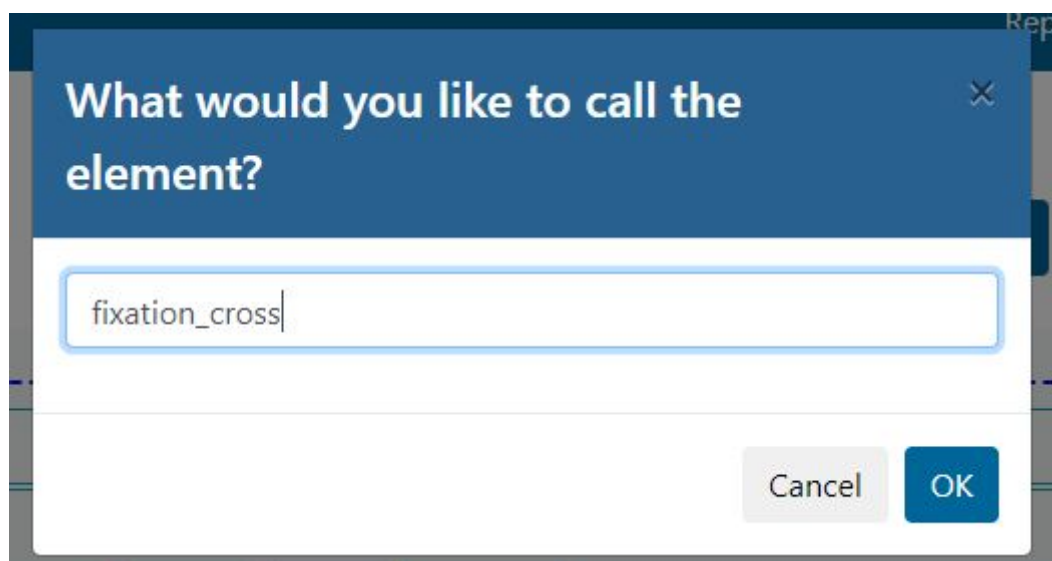


The **dotted blue line** means that at least one of the elements has no stop or hide time (i.e. will display until the end of the trial). Pay attention to how the timeline changes as we change the show and hide times to the stroop word.

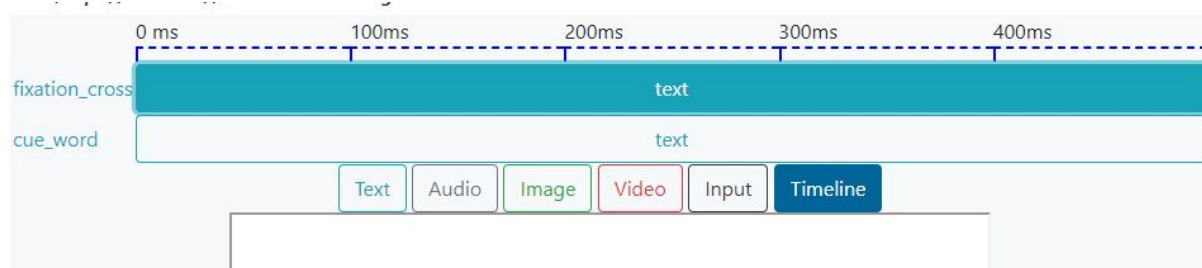
But first, let's change the element names so that the timeline makes more sense. Just click on the element name:



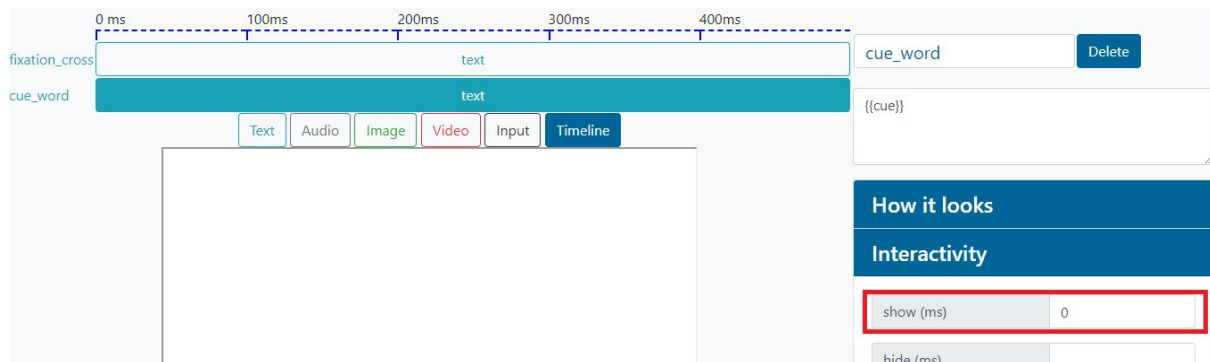
And you'll be asked for a new element id. I'm going to change the fixation cross element's name to "fixation_cross":



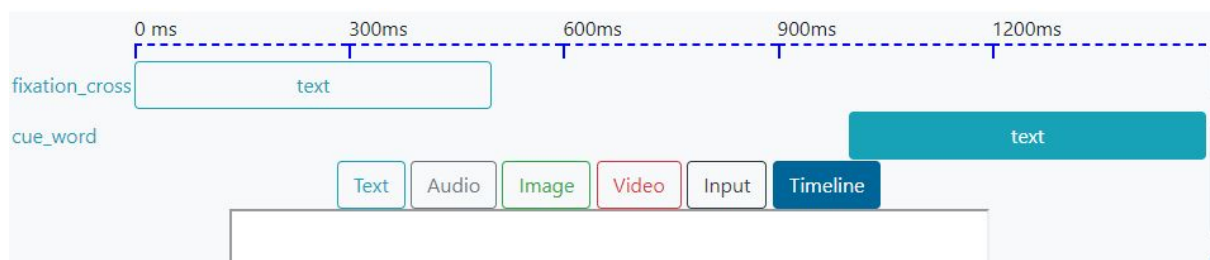
I'm also going to change the stroop word ("{{cue}}") element id to "cue_word". To switch between elements, either click on them in the timeline, or click on them on the **canvas**. The timeline is a bit easier to read now:



The next step is to have a 500ms gap between the fixation cross end and the start of the cue word. So let's just **show** the cue word at 1000(ms). This option is just above the **hide** input we just updated:



Let's change that to 1000 and see what it does to our timeline:



That seems sensible enough. Now we need to actually capture the participant's response!

Capturing response with keyboard

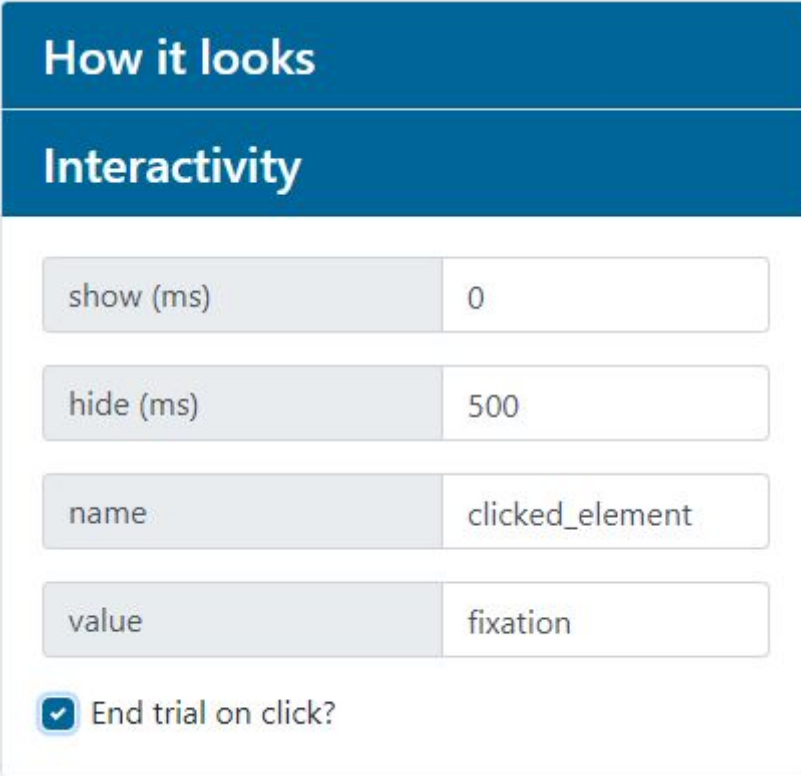
For stroop tasks, you typically need to capture a participant pressing on a key that represents a color. For simplicity's sake, I'm going to accept keyboard responses of "a" and "b". To be able to set the valid keyboard responses click on the canvas anywhere that there isn't an element. You should see the following:



I'm just going to add the letters "g" for green and "b" for blue to **keys**. Notice also the **End trial on key press** checkbox. This is what it says - if you have this ticked then as soon as the participant presses any of the keys you put in the **keys** input, the trial will end.

Capturing the response with the mouse

This is less relevant for a stroop task, but if you want to see what the participant clicked on in the trial, then select that element, go to the **interactivity pane** and change the **name** and **value**. The **name** will be related to the column header in your data sheet. So I'm going to make the name "clicked_element". The **value** will be what value you will put in that column if the participant clicks on that element. In this case I'm going to change the **value** to "fixation". I'm also going to tick the **End trial on click** option. If you do the same you should see the following:



How it looks

Interactivity

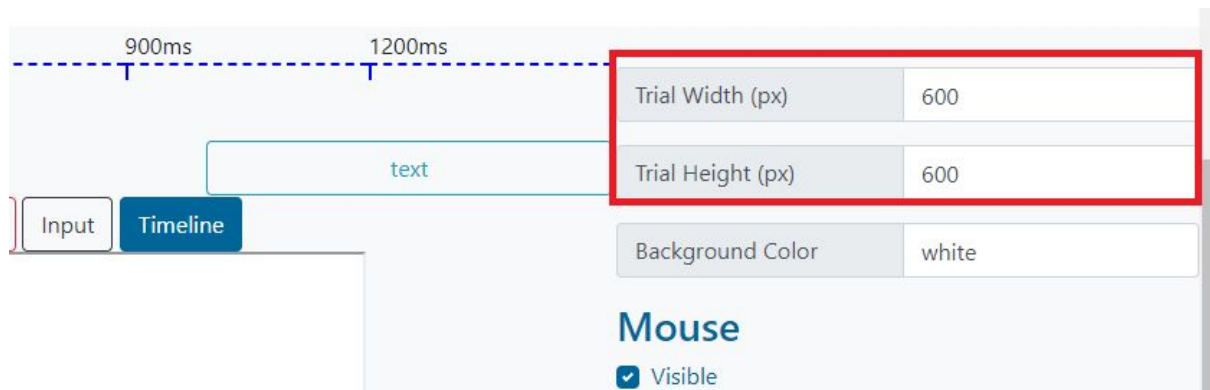
show (ms)	0
hide (ms)	500
name	clicked_element
value	fixation

☒ End trial on click?

Do the same for the cue_word element, but use the **value** "cue".

Changing the canvas size

The canvas, i.e. the space that stores your trialtype, will by default be 600x600 pixels. If you want to change that, just click anywhere on the canvas where there isn't an element, and change the **trial width (px)** and **trial height (px)** settings.



Now we're ready to use this trialtype in our experiment.

Code editor

(Trial type html files can be edited directly using editors like Atom and Notepad++ in the installed version of Collector - go to the User folder)

Saving responses

To save responses in Collector, just give the input with the response you want to save a "name". For example:

```
<input type="hidden" id= "hidden_input" name= "csv_column_header"/>
```

Would put whatever value was in the hidden input **at the end of the trial** into your csv into a column with the header "csv_column_header".

Default javascript and css packages

Your trialtypes start with the following packages without you needing to include them:

```
jquery-3.2.1  
popper.1.12.9  
bootstrap.4.0.0  
bootbox.4.4.0  
PapaParse.4.6.1  
Chart.js.1.0.2  
jstat.1.7.1  
  
style  
jqueryui.1.12.1
```

Collector functions

Whilst the aim in Collector is to try to make it so that your html code for trialtypes would work well anywhere, i.e. are not too reliant on Collector specific functions, there are some exceptions to this where Collector specific functions are either helpful or necessary. These functions should appear in the Collector trialtype code editor when you type "Trial". The functions are:

- ***Trial.add_response(response_object)***: If you would like to add a response row, write what you would like in that row as a javascript object, and put that in as the **response_object** input. For example, I could write something like:

```
Trial.add_response({
  something_to_store: "store me",
  Something_else_to_store: "also store me"
});
```

- ***Trial.elapsed()***: The time since the trial began. This might be helpful if you want to capture the time a participant responded without ending the trial at that point.
- ***Trial.go_to(proc_row_no)***: This will take the participant to the **procedure row** that you have identified with the **proc_row_no**. **Note: the first trial will generally be on row 2:**

Procedure

procedure_1.csv ▾		New Sheet		Rename Sheet	
1	item	trial type	max time	text	Shuffle 1
2	2	instruct	user	1	off

- ***Trial.set(variable_name,variable_content)***: Sometimes you will want to store a variable in one trial, but retrieve it in another. Arguably the easiest way to do this is just to store the variable outside the trial by storing it in the **parent** space. You can do this by adding "**parent.parent**" before any trial that you want to store for future trials e.g:

```
parent.parent.hi_message = "hi there"
```

However, if you don't want to do this, Collector allows you to store variables using **Trial.set(variable_name,variable_content)**. **variable_name** is what you want to call the variable, and **variable_content** is what is stored. For example:

```
Trial.set("hi_message",{ hi: "there"});
```

- ***Trial.get(variable_name)***: this is how you retrieve a variable you stored using `variable_set`, e.g.

```
var retrieved_hi_message = Trial.get("hi_message");
```

- ***Trial.submit()***: This ends the trial and moves onto the next one. Note that if you use ***Trial.go_to*** that you don't need to use ***Trial.submit()***.

Procedure sheet

(Procedure spreadsheet files can be edited directly as a spreadsheet, e.g. using Excel, in the installed version of Collector - go to the User folder)

The procedure sheet describes what order things will happen in your experiment. By default the first trials/events will be at the start of your spreadsheet, and the last will be at the bottom of your spreadsheet. For example, in the procedure sheet below:

Procedure

procedure_1.csv ▾		New Sheet		Rename Sheet		
1	item	trial type	max time	text	shuffle 1	
2	2	instruct	user	This is the start	off	
3	3	instruct	user	This is the end	off	
4						

The participant would be presented with an instruction of "This is the start". When they moved onto the next trial they would get an instruction of "This is the end".

Shuffle 1, 2, 3 etc.

You may want to randomise the order of some of the trials. If you want to do this then you just put the same word into all the rows you want to be randomised together. For example, in the following procedure I want to randomise the order of the middle three instructions:

Procedure

procedure_1.csv ▾		New Sheet		Rename Sheet		
1	item	trial type	max time	text	shuffle 1	
2	2	instruct	user	This is the start	off	
3	3	instruct		Random 1	instruct_shuffle	
4	4	instruct		Random 2	instruct_shuffle	
5	5	instruct		Random 3	instruct_shuffle	
6	6	instruct	user	This is the end	off	
7						

You can see above that I set the value in the “shuffle 1” column to “instruct_shuffle” for rows 3,4,5. Each participant will start with the instruction “This is the start”, and end with “This is the end”, but the order of the middle instructions will be randomised for them. We’ll replace these later with the “stroop_example” *trial type* we’ll create later.

shuffle 2, 3, 4 etc.

Whilst shuffle 1 allows you to shuffle trials within **blocks**, sometimes you’ll want to shuffle the order of blocks. To do this, create a shuffle 2 column to shuffle/randomise the order of the **blocks**.

Collector will use the values written in these columns to randomise the order of blocks. For example, if your procedure sheet looks something like:

Procedure

procedure_1.csv ▾		New Sheet		Rename Sheet			
1	item	trial type	max time	text	Shuffle 1	shuffle 2	
2	2	instruct	user	never shuffle m	off	off	
3	2	instruct	user	block 1 trial 1	block_1	half_1	
4	2	instruct	user	block 1 trial 2	block_1	half_1	
5	2	instruct	user	block 2 trial 1	block_2	half_1	
6	2	instruct	user	block 2 trial 2	block_2	half_1	
7	2	instruct	user	block 3 trial 1	block_3	half_2	
8	2	instruct	user	block 3 trial 2	block_3	half_2	
9	2	instruct	user	block 4 trial 1	block_4	half_2	
10	2	instruct	user	block 4 trial 2	block_4	half_2	
11							

Then because of column “**shuffle 2**” your participant will start with trials on rows 3:6 half the time (on average), and on rows 7:10 the other half (on average).

At the moment you can only shuffle blocks once, but future releases of Collector will let you have a “shuffle 3”, “shuffle 4” etc. to do more block shuffling.

Trial type

The above examples are of the “instruct” **trial type** - which is just a simple way of giving the participants instructions that you write in the “text” column in your procedure sheet. You can create your own **trial types** using the trial type editor at the bottom of the interface.

Alternately, you can copy existing trial types that other users have made from <https://collectalk.com/>.

However, if you followed the above instructions, we now have a “stroop_example” **trial type** we can use. So let’s replace “instruct” with “stroop example”:

Procedure

procedure_1.csv ▾		New Sheet		Rename Sheet	
1	item	trial type	max time	text	Shuffle 1
2	2	instruct	user	This is the start	off
3	3	stroop_exempl	user	Random 1	instruct_shuffle
4	4	stroop_exempl	user	Random 2	instruct_shuffle
5	5	stroop_exempl	user	Random 3	instruct_shuffle
6	6	instruct	user	This is the end	off
7					

The text “Random 1” etc. won’t actually do anything in this case, so feel free to delete it.

Item

In the above example, you might have noticed the “item” column has numbers 2,3,4,5,6 in it. This is because your procedure sheet can refer to stimuli you describe in your **stimuli** sheet. These numbers refer to **rows** 2,3,4,5,6 in the **stimuli**. We start at 2 because when you edit the stimuli sheet here (or in Excel) the first stimuli will be on row 2. We’ll change what’s in the **stimuli** sheet later to refer to **variables** needed for a stroop task.

Max time

This is the maximum time in milliseconds from the start of the trial. You might find that you want your timings to be dependent on events within the trial. In which case you’ll want to use the **code** editor for **trial types**, or perhaps a **boost** that you apply to a **trial type** that controls timings in this way.

Text

This is only relevant for trial types that include a “{{text}}” **variable** in their code. The instruct **trial type** does, so it’s an easy way to add instructions. In the **procedure** spreadsheet.

Weight/freq/frequency/repeat - useful for piloting your task

Weight (or **freq** or **frequency** or **repeat**) sets how many occurrences of a trial you will have. By default, you will only have one occurrence of a unique trial (see **Item** for how to refer to multiple rows in the stimuli sheet). If you set the **weight** to 2, then you would have twice as many occurrences of the trial(s) as normal. **If you set the weight to 0, then you will skip that row in the procedure sheet. This can be useful for piloting parts of your task.**

Stimuli sheet

(Stimuli spreadsheets files can be edited directly as a spreadsheet, e.g. using Excel, in the installed version of Collector - go to the User folder)

This is where you define the stimuli or variables for each trial. In our stroop task, we need to update the “cue” column to reflect the words used in a stroop task, and add a column “letter_color” which defines the colors of the word. So let’s just have 4 trials for each combination of blue and green:

Stimuli

stimuli_1.csv ▾	New Sheet	Rename Sheet	List Stimuli
1	cue	answer	letter_color
2	blue	Apple	blue
3	green	Banana	green
4	blue		green
5	green		blue
6			

You may notice there’s a redundant “answer” column - that’s fine, it’ll just be ignored.

List Stimuli button

If you are running an experiment with pictures, videos and/or audio files then there’s an easy way to generate a list of all the stimuli if you’ve uploaded them to the **stimuli** folder on dropbox. Just click on the **list stimuli** button below the **Stimuli** title:

Stimuli

stimuli_1.csv ▾	New Sheet	Rename Sheet	List Stimuli
1	cue	answer	letter_color

You should get an interface like this:

2 out of 2

All

None

Filter

filter

Apply

Spreadsheet

/stimuli

CollectorCat.png

kitten.png

It might take a minute or two, depending on how many stimuli files you have.

Spreadsheet button - If you click on the “Spreadsheet” button, you should be able to download a list of the files (and **very importantly** their dropbox locations). This can save time massively if you have a lot of stimuli.

Filter input - if you have lots of stimuli, just type in letters for what you want to keep. For example, if I only wanted “CollectorCat.png”, I could type in “Cat” (**This IS case sensitive**) and then click **apply**. I would then see:

2 out of 2

All **None** **Filter** Cat **Apply** **Spreadsheet**

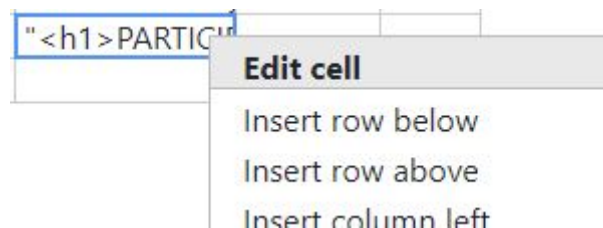
/stimuli CollectorCat.png

Importantly, if I clicked on the **spreadsheet** button now I will only get information for the filtered list.

Cell editor

This can be extremely helpful.

When editing spreadsheets in Collector, there may come a time when you want to have a lot of text within a cell. For example, if you'd like to use [surveys](#) to create an information sheet and/or consent form, you might want a lot of your text to be in a single cell. To allow you to edit a single cell with a lot of text more easily, right click on it and select **edit cell**:



You should then see something like this:

Add
s
B
I
U
H
↓
Size

Auto-Preview (light):
On
Off
Full Preview

```

1 <h1>PARTICIPANT INFORMATION SHEET </h1>
2
3 <h2>Title of Project: Your project title.</h2>
4
5 <h3>Name of Lead Investigator: Your name </h3>
6
7 <h4>Invitation paragraph</h4>
8 Here's a bunch of text.<br>
9
10 <h4>What is the study about?</h4>
11 Here's a bunch more text.<br>
12
13 <h4>What does the study involve?</h4>
14 More and more text. <br>
15
16 <h4>Do I have to take part?</h4>
17 Explanation that the answer is essentially no.
18
19 <h4>Contact for Further Information</h4>
20 Contact info here

```

PARTICIPANT INFORMATION SHEET

Title of Project: Your project title.

Name of Lead Investigator: Your name

Invitation paragraph

Here's a bunch of text.

What is the study about?

Here's a bunch more text.

What does the study involve?

More and more text.

Do I have to take part?

Explanation that the answer is essentially no.

Contact for Further Information

Contact info here

The left side is where you edit what goes into the cell, the right side is a preview of what it might look like in your experiment. Some key points:

**Add
s:** If you've copied and pasted a bunch of text from another file (e.g. word), you may need to click on this to make sure that the text is spread across different lines of text in the way that you would like them to be. For example, if I copied:

This is the first line of text
This is the second line of text
This is the third line of text

I would get:

Add
s
B
I
U
H
↓
Size

Auto-Preview (light):
On
Off
Full Preview

```

1 This is the first line of text
2 This is the second line of text
3 This is the third line of text

```

This is the first line of text This is the second line of text This is the third line of text

Which is not what I want. However, when I press **Add
s**:



(note the `
`s that have been added at the end of each line)

B(oid), I(talics), U(nderline): These don't work quite the way as a normal text editor, but will generate the code needed to allow you to start writing in bold or italics or underline, or convert whatever you had highlighted accordingly. If you want to stop writing something in bold, for example, just press the right arrow until you've moved past the code that is making the text bold (e.g. `YOUR TEXT`). This can be a nice way to learn some basic HTML and css code.

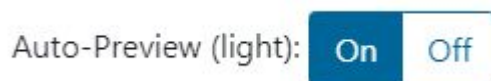
H(eader): Like the above, this will create a Header using an `<h2>YOUR HEADER</h2>` tag. Play around with the h numbers (e.g. `<h1>`, `<h3>` etc.) to see how big you want your header to be.



This just allows you to change the color of your text, similar to the above.



This allows you to change the size of your text. By default, it'll assume you mean "px" or pixels, but you can explicitly say this, e.g. "12px".



This is on by default, so that you get live updates as you edit the cell contents. However, if you have a really really large cell, you might want to turn this off if your browser is slowing down. (I haven't experienced this yet though...)



Auto-preview gives a simplified preview, but doesn't use the default libraries that Collector uses. Click on Full Preview to see what the content should look like in practice. (Having this on all the time is **much** slower than the light auto-preview).

Helper sidebar

On the right hand side there is a “helper” sidebar. You’ll notice that the advice it gives you changes depending on what you’re focusing on. For example, when I’m looking at the “participant_id” column in the **conditions** sheet I get the following help:

participant_id	end_message	start_message	buffer	
random			5	

To see the whole cell when editing the sheets, either press F2 or double click on the cell

participant_id

Participants give their ID

By default, participants will give a **participant id** (e.g. a Prolific Academic or MTurk ID) at the start of the experiment. Change this setting to either **random** (recommended) to generate a unique and randomised id for the user, or **off** (not recommended) for the participant to have the id 'notUnique'. The participant_id influences the name of their data file. If the participant does not put in an ID, they will skip the **start_message**.

General tips

Downloading and Uploading experiments

On either side of the experiment name, there’s an up arrow for uploading an experiment and a down arrow for downloading the selected experiment:



This can be an easy way to duplicate an experiment. Just download it, and then upload it and give it a different name!

If your experiment included trialtypes that were written using the **Code** editor (as opposed to the **Graphic** editor), Collector should ask you if you want to upload those trialtypes also. Similarly, any Surveys that were included in the original experiment should be downloaded and uploaded accordingly.

Running your study

Save!

Make sure you've saved! You usually can just press CTRL-S to save. The best way to save all your progress is click on the save button in the top left of the **studies** interface:



Conditions

1	name	notes	stimuli	procedure	fullscreen	welcome
2	condition_1	You can put mo	stimuli_1.csv	procedure_1.cs	off	
3						

Both Online and Installed versions

There are some differences between the online version (e.g. on <https://ocollector.org> or <https://some-open-solutions.github.io/ocollector>) compared to when you are running Collector locally on your computer after having installed it.

Click the “run” button

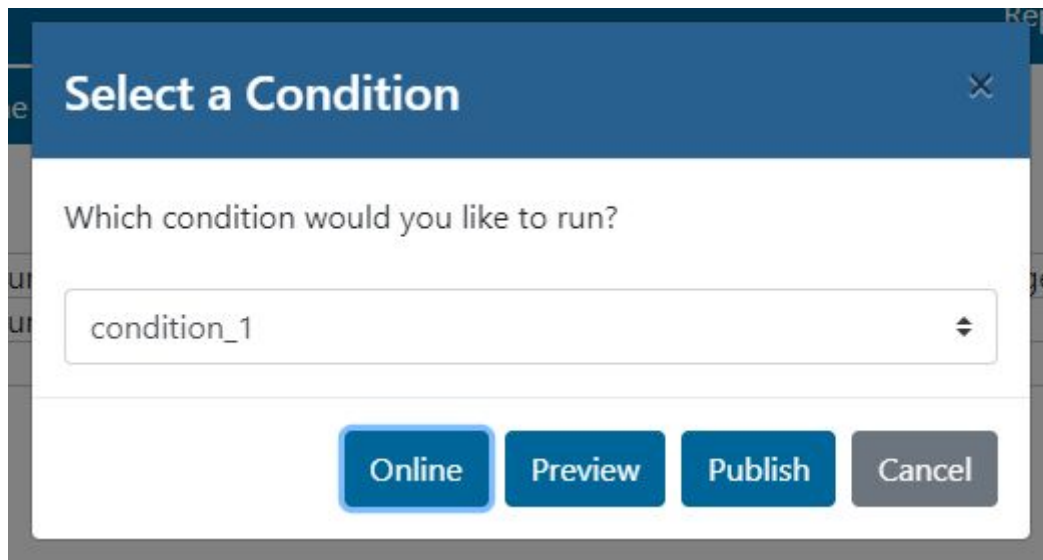
When you click on the **run** button:



Conditions

1	name	notes	stimuli	procedure	fullscreen	welcome	participant_i
2	condition_1	You can put mo	stimuli_1.csv	procedure_1.cs	off		on
3							

You'll generally see an interface like the following:



Online - will take you to where you would want to send participants. But see [publish your task](#) before sending participants this link. Also, if you are using the installed version, [you will need to update your github website by clicking on the github button](#).

Preview - this should run the task without saving data


Publish - this will allow you to publish your task onto a server that will handle either saving

of your data. You can also get to the **publish** interface by clicking on the **C** icon in the top right of the screen.

Publishing your task

After clicking on **run** → **publish** or just the **C** icon on the top right of the screen, you should see something like:

Register

Server	Register User	Register Experiment
ocollector.org	Create Account	Publish
	Delete Account	Unpublish
	Forgot password	
open-collector.org	Create Account	Publish
	Delete Account	Unpublish
	Forgot password	
Add server		
<div>email</div> <div>password</div> <div> <input type="checkbox"/> I'm not a robot <div>  <div>reCAPTCHA</div> <div>Privacy • Terms</div> </div> </div>		

If you are a member of one of the following universities, you are automatically able to have your data e-mailed to you:

- University of Reading (i.e. have an @reading.ac.uk e-mail address)
- University of Reading Malaysia (i.e. have an @reading.edu.my e-mail address)
- University of California, Los Angeles (i.e. have an @g.ucla.edu e-mail address)
- University of DeMontford (i.e. have an @dmu.ac.uk e-mail address)

If you are **not** a member of one of the above institutions you will need to complete the [Saving data with google drive](#) step before proceeding.

To get your data e-mailed to you and backed up, put in your e-mail and password and then click on **create account** for the server you want to register with. Once you have successfully registered with that account, you can then **Publish** your experiments to that server. Once you've successfully published your experiment, you're ready to start data collection!

Note, if you publish your experiment with multiple servers Collector will use some of the servers as back-ups in case the first server fails.

Once you've reached here with the stroop task (and have successfully registered your experiment), close this interface and click on the **run** button as described above. You can now click on the **online** button to see how your task looks (and allow us to look at your data in a bit).

Saving for installed version only

You don't need to read the following if you are using Collector purely online, e.g. on <https://ocollector.org> or <https://some-open-solutions.github.io/ocollector>. If you have installed Collector and are running it on your machine, you will need to **push** your changes onto the Collector Repository you set up.

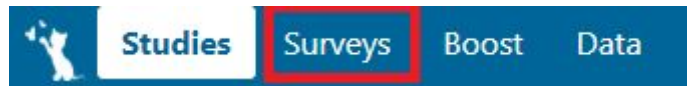
Have multiple studies in a row

You might find that trying to have everything your participant does in one experiment is a bit too complicated to fit into one procedure sheet and stimuli sheet. A way to deal with this is to break your research up into multiple experiments. To make the participant move from one experiment to another, add a "forward_at_end" column to your conditions sheet, and then put the URL of the next part of your experiment in there.

To find out the URL, open the experiment in question and run it, selecting the "online" option.

Surveys

To create and edit surveys, go to the Survey tab at the top:



You cannot create a survey from scratch, you need to select a previously existing survey to base the new survey on. I'm going to use demographics just because it has a lot of different types of questions. Just select "demographics.csv" and then click on **new survey**:



The **helper bar** on the right should tell you a lot about each column depending on where you click in the survey. So I'm just going to highlight the fact that after each change to the survey you can preview you changes by clicking on the **preview** button:

A screenshot of the survey preview interface. At the top is a navigation bar with 'New Survey', 'exemplar_demographics.csv' (with a dropdown arrow), 'Save', 'Rename', 'Delete', 'Spreadsheet', and 'Preview' buttons. The 'Preview' button is highlighted with a red rectangular box. Below the navigation bar are several survey questions and their corresponding answer options:

- Gender: Male, Female, Other
- Age: Text input field
- Education (which level are you currently in, or is the highest you've completed): -- no option selected -- (dropdown menu)
- Do you speak English fluently?: Yes, No
- At what age did you start learning English? (write "0" if from birth): Text input field
- What is your country of residence?: Text input field
- What is your preferred hand?: Left, Right, Both
- If you would like to please state your ethnicity: Text input field

If you want to go back to editing, just click on the Spreadsheet button to the left of the preview button:



Type

There are a lot of different **types** of survey question you can have. If you click on any cell in the “**type**” column you should see a summary of each type of question:

checkbox

If you want the participant to be able to select multiple options, use a checkbox. If you only want them to be able to select one option, use a 'Radio',

date

This allows standardised storage of date. Note that this works better in Chrome than firefox.

dropdown

A standard dropdown list.

instruct

Allows you to type in instructions.

likert

Participants can click on one of the options that they want.

number

Only allows participants to type in a number

para

Gives the participant a large text box to write in their answer. ",

radio

Allows the participant to only select one option in a list (unlike checkboxes)

shortAnswer

Gives participant space for a small paragraph for their response

slider

The participant can slide between values you specify

text

Gives the participant a text box to write in their answer

answers

Answers the participant can choose

A single answer or list of answers the participant can choose. Put a pipe character (|) between each possible answer. If you're having problems finding the pipe key, you can highlight the pipe in the brackets and then copy and paste it.

values

What each answer is worth

Values work in parallel to **answers**, in which you associate a score for each answer. For example, if the answers the participant could choose were 'strongly agree|agree|disagree|strongly disagree' and the values were '3|2|1|0', then the value of strongly agree would be 3, agree would be 2, disagree would be 1, and strongly disagree would be 0. These value of the participant's response is stored by itself in the data, and can also be used when adding together the **score** across multiple items. To get the score across multiple items, create a column with 'score:[your score name]'. Once you've created this column, click on any cell in that column to get more guidance.

score:[your score name]

What is the scale or subscale you want this questionnaire to contribute to

To get a score for a questionnaire, you need to have a column header that is formatted with "score:[name of your scale or subscale here]". Inside this column you can add a "1" or "r1" for each question you want to contribute to this scale (see answers and values above). 1 will add the question to the score as described in the **answers** and **values** columns and "r1" will do the same but with the **values** reversed.

feedback

Tell the participant about each answer

You can list what feedback you associate with each response. Put a pipe | between each feedback

feedback_color

What color is each feedback

The colors you want to associate with each feedback (e.g. green for the correct ones, and red for the wrong ones. Put a pipe | between each feedback color

item_name

Column name in data sheet

The **item_name** will be what the column header is in your data sheet. So make sure it's unique!

no_text

No text on left side of survey

If you have an item, in which you want the content that would normally be on the right side of the survey to take up the whole width of it, put **'on'** in this column.

optional

Can a participant ignore this item?

The default assumption is that a participant doesn't need to respond to any particular item in your survey. If they need to reply to an item then put a **no** in this column.

shuffle

Randomise the order of items

In this column, if you want to randomise the order of questions, you can either simply write **'on'** in the items you want randomised, or you can use a different words for different groups of items you want randomised together. For example, if you wanted the first and second half of your items shuffled, you could write 'first' in the shuffle column for each of the first half of the items, and 'second' for each of the second half of the items. The order of items is preserved otherwise, so any items without a word in the shuffle column will not be shuffled. If you write 'off' in this column, the item will **not** be shuffled.

text

Question text

This will be presented on the left side of the survey. This is useful if you have a question with responses for this item. If you want to have no text for this row of the survey, create a column **no_text** in your sheet, and turn it on. If you want the text to take up the whole row, then under the **type** header, choose 'instruct'

Data

If you are one of the institutions supported by Some-Open-Solutions, then you can skip the “Saving data with google drive” step to store your data. The institutions currently supported are:

- University of Reading (UK and its Malaysia campus)
- University of DeMontford (UK)
- University of California: Los Angeles

Saving data with google drive

You can set up data storage on a google drive for free. Before proceeding, you may want to consider:

- whether you plan to store any **personally identifiable data!**
- If you have to comply with data protection rules, e.g. GDPR

Some things to bear in mind before starting the process of setting up a “server” with google scripts:

- The script will save data at the end of the experiment. **This is because saving after every trial could quite quickly get you to your 50,000 script calls a day hard limit.**
- However, if you are confident you won’t use over 50,00 script calls within a day, you may want to save after every trial. This is what I’ll suggest you do for eye-tracking (described at <https://github.com/Collector-Trialtypes/webgazer-eyetracking>).
- If a participant stops part way through, their data is available to them on their computer. They can save their partial data by pressing CTRL-S and then email it to you.
- Again, think carefully about whether google drive is an acceptable solution for storing data. Avoid storing personally identifiable information on it as regulations like GDPR require user data to be stored in specific locations like European servers. If you would like more control on where your data is stored, then [Saving and emailing data with a server](#) might be what you want to do.

If not, then you’re fine to proceed with storing participant data on a google drive. If so, you may need to store data on a server that is compliant with the regulations for your area. Whilst Collector encrypts the participant data on their device, data protection regulations can demand personally identifiable data is stored on servers in specific locations (e.g. Europe for GDPR). You can hire relatively inexpensive server space for less than £100 a year, so go to [Saving and emailing data with a server](#) for some instructions how to set your server up.

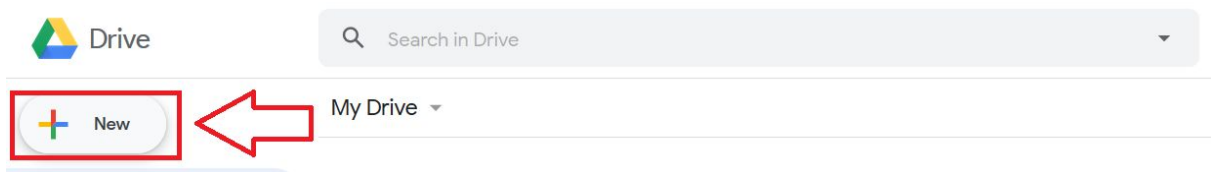
However, to save data to google drive you can click on the **google script** button in the data tab:



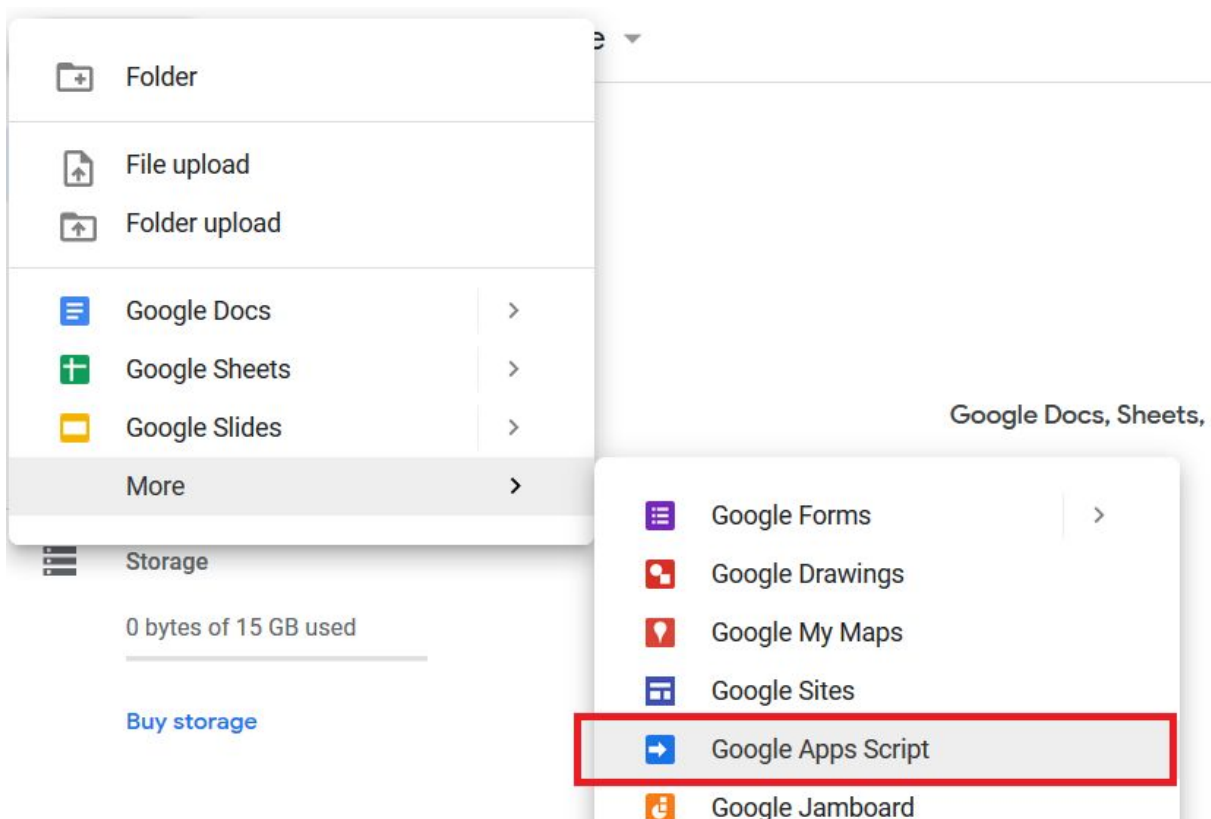
This should present you with a script you can save to google drive.

Now let's open up google drive. You may well have set up a google account for using Collector online, and this would be a good account to store your data in. **It is strongly recommended that you DO NOT use a personal account for storing data. The google script you will be putting onto google drive writes files, which is something you don't want happening on your personal account!!!**

So if you haven't yet got a google account for saving participant data, now would be a good time to create one. Once you've created it, go to **google drive**, and you should see:

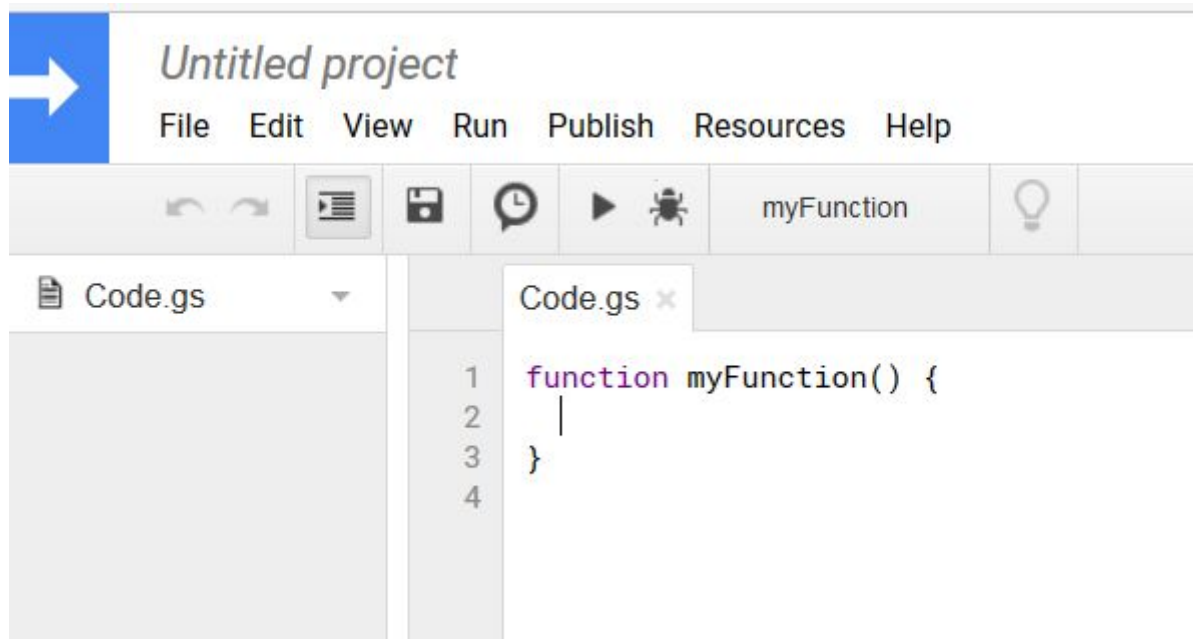


Click on the **New** button and then select **Google Apps Script**:



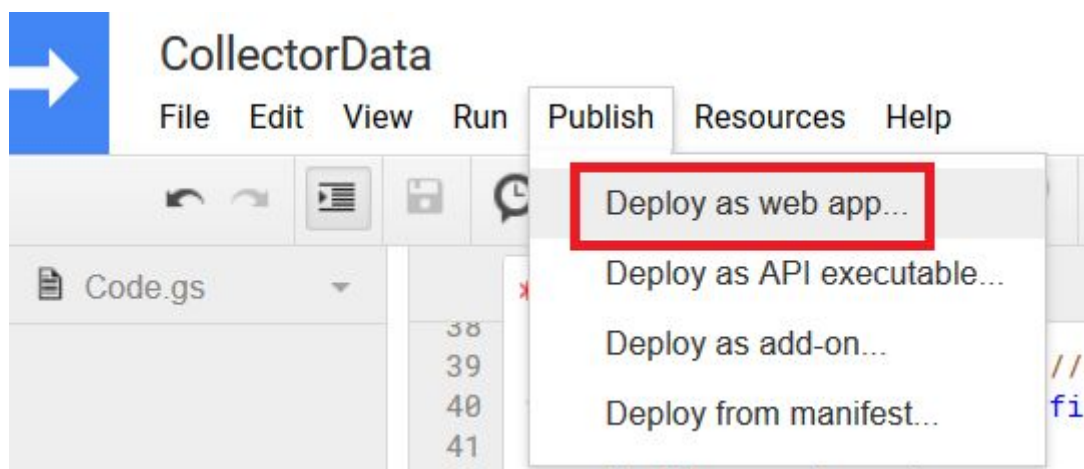
You might have problems proceeding to the google apps script **if you're already signed in with a different google account**. One way to deal with this is to open google drive on a new browser in which you're not signed with any google accounts.

Once you're editing the script you should see something like:



Delete everything in the script, and then paste the script you copied earlier from the **google script button**. Then Save (Ctrl-S works to do this on windows). You'll be asked for a project name, I use "CollectorData" but I don't think it actually matters.

Now that you have a saved script, you just need to publish it to get a link that Collector can use! To publish it, click on **publish** and then **deploy as web app...**



There are some important settings to make sure your script works as you intend it to:

Deploy as web app

Project version:

New
Describe what has changed...

Execute the app as:

Me ()

You need to authorize the script before distributing the URL.

Who has access to the app:

Anyone, even anonymous 

[Help](#)

Make sure that you are executing the app as yourself (i.e. leave this as 'Me'). Make sure to change the **Who has access to the app** setting to "Anyone, even anonymous", because that is who your participants are. And this is how their data is written to your google drive.

Once you've fixed the settings, you can click on **deploy**. You will then be asked to authorize this script. **The next steps illustrate why you do not want to be doing this on your personal gmail account!**

Click on the **Review Permissions** button when you are asked, and then you should see be asked to confirm your google account. After you've done that you should see:



This app isn't verified

This app hasn't been verified by Google yet. Only proceed if you know and trust the developer.

Advanced

BACK TO SAFETY

To get past this, you need to click on **Advanced** and then click on **Go to [whatever you saved the script as] (unsafe)**



This app isn't verified

This app hasn't been verified by Google yet. Only proceed if you know and trust the developer.

[Hide Advanced](#)

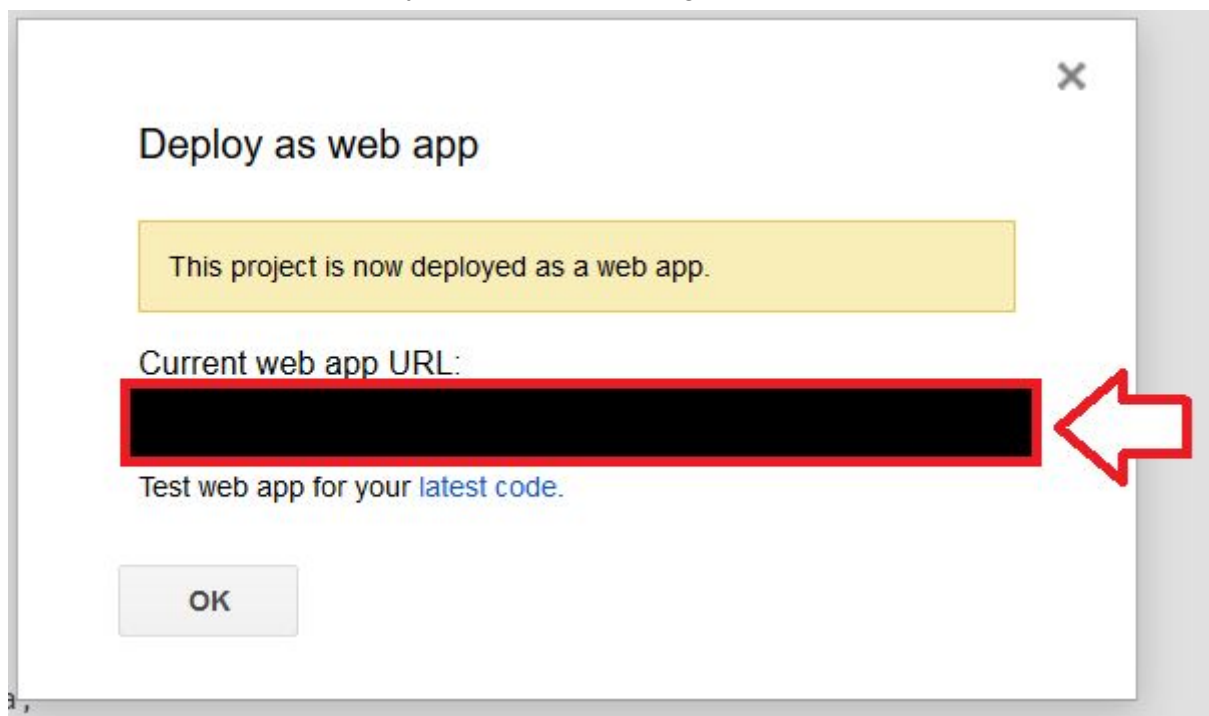
BACK TO SAFETY

Google hasn't reviewed this app yet and can't confirm it's authentic. Unverified apps may pose a threat to your personal data. [Learn more](#)

[Go to CollectorData \(unsafe\)](#)

Then you can click on **Allow** to proceed.

You should now be able to copy and paste the URL generated:



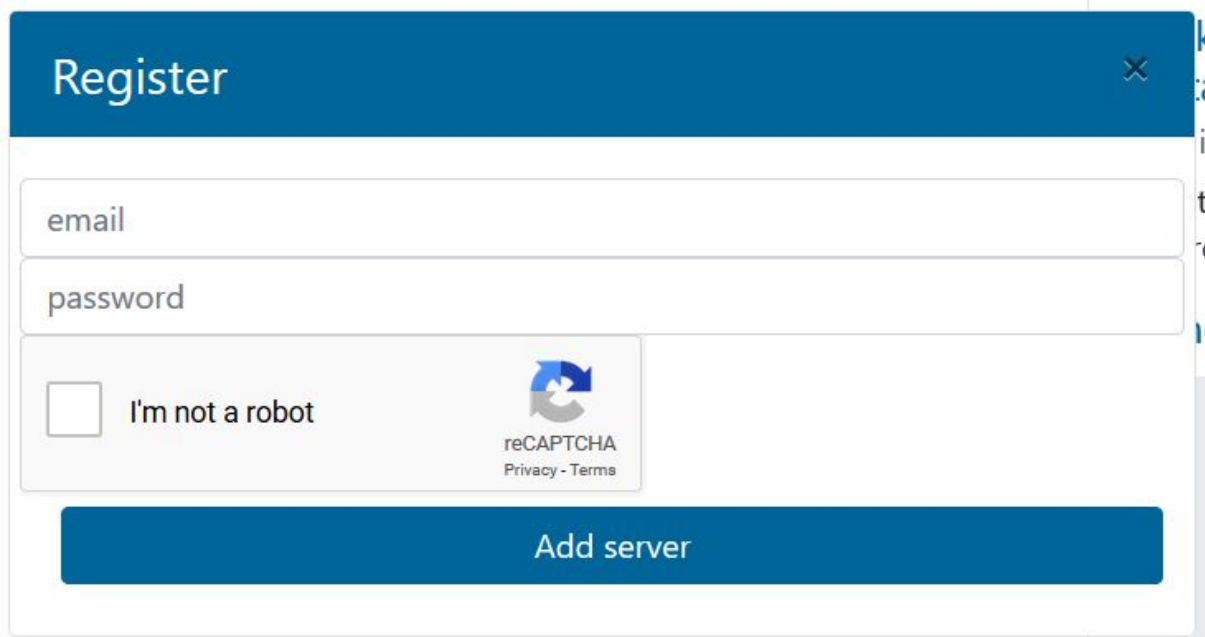
to Collector to save your data into google drive.

The next is [Telling Collector where your saving/emailing script is](#).

Saving and emailing data with a server

Telling Collector where your saving/emailing script is

If you have set up a script for saving your data as described above in [Saving data with google drive](#) or [Saving and emailing data with a server](#), then you need to tell Collector where to find these scripts. To do this, click on the Collector Icon in the top-right corner of the screen to get the following:



The image shows a 'Register' dialog box with a blue header bar containing the title 'Register' and a close button (X). Below the header are two input fields: 'email' and 'password'. Under the 'password' field is a reCAPTCHA widget with a checkbox and the text 'I'm not a robot'. To the right of the checkbox is the reCAPTCHA logo and the text 'reCAPTCHA Privacy - Terms'. At the bottom of the dialog is a large blue button labeled 'Add server'.

Click on the **Add server** button to add your script. You'll first be asked for a script to save the participant data, where you put the script described in [Saving data with google drive](#) or [Saving and emailing data with a server](#). You will then be asked if you want to add a script for registration. This will only be relevant if you completed the steps described in [Saving and emailing data with a server](#). Either add in a script for users to register with, or

Decrypting your data (and understanding it)

I'm going to include some clarifications about actually running your task in this section, so feel free to skip to [decrypting your data within the "Data" tab](#) if you already have been e-mailed or downloaded data.

If you're working through the stroop task exercise, you'll see something like the following when you run your experiment:

Collector

It's very important to read the following before starting!

If you complete multiple Collector experiments at the same time, your completion codes may be messed up. Please do not do this!

If you participate in this experiment, your progress in it will be stored on your local machine to avoid you losing your progress if the window or tab closes or freezes. This data will be cleared from your computer once you have completed the task. **However, if you do not want this website to store your progress on your computer, DO NOT PROCEED.**

If the experiment freezes, try pressing **CTRL-S** to save your data so far.

I'm just going to run the experiment as "example_p".

Assuming everything went okay, you'll get through the task until you reach:

Please wait while we confirm that all your data has been saved

After a while, the data should be saved, and you'll see:

Thank you for participating. If you'd like to download your raw data [click here](#)

This is totally optional, but please select the country you are completing this from and then click on "Submit"

Submit



It seems that the lag between completing an experiment and the researcher's e-mail acknowledging it vary quite a lot. However, if your participant sees the second screen with the map the data is on the way. If they get stuck on:

Please wait while we confirm that all your data has been saved

...for 5 or more minutes then something has gone wrong. You may need to ask them to press CTRL-S to save the data so that they can email it to you directly if this occurs.

Decrypting your data within the "Data" tab

Once your data has arrived in your e-mail, download it, and then go to the **data** tab at the top of the Collector interface:



Once there you can click on the "Decrypt files" button, select your file in your downloads folder (or wherever you saved it).

You'll need the password you created originally to decrypt it. Once you've typed it in you should be presented with your file!

Understanding your data

If you've been working through the stroop example, your data should look something like this:

P	Q	R	S	T	U
cue	answer	letter_color	keyboard_clicked_element		
blue	Apple	blue			
blue		green	g		
green		blue	b		
green	Banana	green	g		

You can see in column **s** "keyboard_response" that I pressed g then b then g. This seems to be correct as "b" represents me thinking the letters were blue, and "g" represents me thinking the letters were green (which matches column R "letter_color"). Had I clicked on the fixation cross or the word, then column **t** "clicked_element" would have included the words "fixation" or "cue".

FAQ/General questions

Can I use Collector if I am not a programmer?

YES! This program was designed to allow researchers to work with a format they're very familiar with (spreadsheets) to create interactive experiments. As you will see in the tutorial Collector is almost a completely programming free solution. Of course there are times when being able to code will make your life easier but we have tried to minimize user coding wherever possible.

What language is Collector written in?

Collector is written mostly in javascript/Jquery. Formatting of the presentations is controlled mostly by HTML5 and CSS2.

Do I need to ask anyone if I want to use Collector?

No. Collector is distributed under the GNU GPLv3 license (full text can be found in the **Web/Admin** folder). You are free to use, modify, and distribute this program as you wish. If you distribute a modified version of this program you are required to make your modified version available to the public under the same GNU GPLv3 license of the original program. If you are publishing papers that have used Collector experiments we ask that you acknowledge its use somewhere. This isn't a requirement of using this software but we'd like to get the word out to as many people as possible about its availability.

Collector will be citeable in a prepublication journal later this year.

If you build trial types that you think others would find useful please go to <https://www.collectalk.com> to share.

Troubleshooting

My changes aren't being reflected in the experiment. Why?

1. Save all your files before running the experiment. 90% of the time when you are having this problem it is because you forgot to save the changes you've made. Seems silly but I run into this all the time.
2. Clear your cache. There are extensions for chrome (<http://goo.gl/r961j>) and firefox (<http://goo.gl/r3Zky>) that make this process very quick
3. Are you sure you're editing the right experiment/folder? (this is for if you edit the experiment files directly, not within the Collector.py interface)

Further notes

What happens with my password for encryption and decryption

Upon creating a Collector folder in your dropbox account, a public and private key are generated using tweet nacl library by dchest available at

<https://cdnjs.cloudflare.com/ajax/libs/tweetnacl/1.0.1/nacl.min.js> . The public key is used for encrypting data, the private key is used for decrypting data. Your private key is encrypted using Crpyto-Js available at <https://cdnjs.cloudflare.com/ajax/libs/crypto-js/3.1.9-1/crypto-js.min.js>

This means that your private key can only be used when you type in your password, and thus the data can only be decrypted by someone who knows your password.

What happens with my password for Collector

When creating an account on Collector you are asked for a password. This password is hashed using a randomly generated salt and pepper at the start and end of your password using PHP:

```
$salt = create_random_code(20);  
$pepper = create_random_code(20);  
$prehashed_code = create_random_code(20);  
$hashed_password = password_hash($salt.$user_password.$pepper,  
PASSWORD_BCRYPT);  
$hashed_code = password_hash($prehashed_code, PASSWORD_BCRYPT);
```

This means that we don't store your password in bare form on the server. Next time you log in your password is compared to the hashed version, and if they match then we accept it is you, and keep you logged in for 30 minutes. If you don't do anything for 30 minutes then we'll ask you to log in again.

In case you're curious - here's how we generate random codes:

```
function create_random_code($length){  
    $characters =  
'0123456789abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ';  
    $charactersLength = strlen($characters);  
    $new_code= '';  
    for ($i = 0; $i < $length; $i++) {
```

```
$new_code .= $characters[rand(0, $charactersLength - 1)];  
}  
return $new_code;  
}
```

Contributing

Any contributions to Collector would be very welcome! The idea when it was initially registered as open-source software was to make it easy to do online research for free if possible, or as cheaply as possible. There are three key ways to contribute:

- Highlight any problems you find with collector or its documentation on <https://collectalk.com>
- Add your own trialtypes (and exemplar experiments) on <https://collectalk.com>
- Contribute to the code for Collector itself! A good starting point is to fork the repository at <https://github.com/some-open-solutions/collector-dev> and drop me an e-mail (a.haffey@reading.ac.uk).

Original contributors

Collector was initially created by Mikey Garcia as software that should allow researchers to conduct online studies for free. The below is taken from the [original repository](#) he created:

Tyson Kerr

"Many of the best ideas/solutions in Collector have come from the mind of Tyson. Tyson's contributions are so wide that it is hard to think of a piece of the code he hasn't been involved with at this point. Despite his broad contributions, I think Tyson would agree with me that his real baby is the getdata functionality in /Data/. Every time you use those slick menus to check participant completion, exclude flagged users, or download all your precious data into one clean sheet you have Tyson to thank."

Victor Sungkhasettee

Figured out how to implement the Audio trial type

Adam Blake

"Completely reorganized the collector.js code to be object oriented. Is responsible for the current look of Collector because he redid nearly all of the CSS to make it much prettier than I initially could."

Nate Kornell

"Without Nate there would be no Collector. Many years ago Nate taught me how to use the tool he had created for himself, Generic, and that code inspired me to write Collector. Most of the core ideas and design decision at the heart of this project are either directly lifted from Nate's program or were based on adaptations of what he had created."