Running Experiments Using jsPsych and MySQL

A short, step-by-step guide to be used with jsPsych's online tutorials.

Ruthie Shaffer

Annoying ambiguity is annoying. Sometimes I just want to know exactly where on the screen to put my mouse to make the right things happen. In this spirit, at several points I've included descriptions that may be overkill for some but will hopefully be helpful for others.

Contributions: Generally, CSE330's online (wiki) textbook is fantastic for many topics. I refer to it often, and much of this manual stems from things I learned in this class/on the class's wiki pages. Further, many aspects of this manual stem from things I learned on jsPsych's documentation pages / forum. For the list of references, as well as some helpful links, see the References section at the bottom of this file.

Note: Anytime you see **<**any_words**>**, this means fill in whatever name applies to the situation (without the **<>** symbols). Also note that some links may need to be copied and pasted in order to work (rather than clicked in this file).

This manual was written with the aim of providing an approachable starting point for online data collection for those with little-to-no programming experience.

Purpose

Provide a starting point for using MySQL with jsPsych to collect data.

Step 1

Set up a local web development environment

This will be done using the AMP stack (Apache, MySQL, PHP).
And this link (for El Capitan):

- https://coolestguidesontheplanet.com/get-apache-mysql-php-and-phpmyadmin-working-on-osx-10-11-el-capitan/
- I've found the above link very useful
- Use the "User Level Root" instructions when given the choice

For *local* web development, you can also download and use MAMP. However, this manual assumes you've set up your own environment using the above tutorial. If you decide to use MAMP, the only real differences would be in Steps 1 and 2 of this manual and relate to loading your content online. If using MAMP, remove "~<username>/" from web addresses below.

Step 2

Get going with jsPsych

Begin here: http://docs.jspsych.org/tutorials/hello-world/

- Complete all 6 steps. Below are a few notes about steps 2, 4, and 5 that may be helpful.
  - For step 2:
    - They instruct you to create a folder.
    - Put this new folder in the Sites folder from the AMP stack tutorial above (or the htdocs folder, if using MAMP)
    - Like they instruct you to, put your jspsych folder download inside this new folder.
    - The directory set up should look like: /Users/<username>/Sites/<experiment_folder_name>/jspsych-5.0.3 (or whatever version of jspsych you have)

- For step 4:
  - When they say to open the file in a web browser:
    - Clicking on the html file will open it in your default web browser, but it won't really be working as it should.
    - The path will look like file:///Users/<username>/Sites/<experiment_folder_name>/experiment.html
    - Instead, type the following into your web browser (don't forget the ~ symbol): http://localhost/~<username>/<experiment_folder_name>/experiment.html
  - If it says you don't have permission to access this:
    - Go to terminal to give yourself permissions to access the experiment folder
    - The AMP stack tutorial above recommends using:
      - sudo chmod -R a+w ~/Sites/<experiment_folder_name>
      - Bear in mind the tutorial says: "To run a website with no permission issues it is best to set the web root and its contents to be writeable by all, since it's a **local development** it shouldn't be a security issue."
- For step 5:
  - Now that you've begun importing files, be sure that your paths are correctly defined:
    - The scripts provided as is point to a folder called jspsych-5.0, and your download might have a different name (i.e. jspsych-5.0.3). So make sure to change this as needed in your experiment.html file to reflect the correct folder name.

Next, complete jspsych's second tutorial (more comprehensive exposure to jspsych) http://docs.jspsych.org/tutorials/rt-task/

- If you don't hate yourself ☺ → Work through the tutorial step by step. Make sure you understand what each step is doing.
- If you hate yourself ☹ → Copy "The Final Code" at the end of the tutorial instead…but you shouldn't do this. The tutorial is relatively quick and very helpful.
- Again make sure to change the paths in your html file from jspsych-5.0 to whatever your jsPsych download folder is named if necessary.

Next get going with jsPsych data storage: http://docs.jspsych.org/features/data/

- Go through and complete all parts of “Storing data in jsPsych’s data structure”, using the experiment file you just made (as a part of the rt-task jsPsych tutorial).
- When implementing “Adding data to all trials”
    - You can simply place the given code at the top of the <script> section (above the welcome_block variable).
    - To see that it worked, run through the experiment in your web browser and see the final data structure the program spits out at the end (you have to press a button on your keyboard to get past the final experiment screen and to the screen that shows the data structure). You should see “subject” and “condition” in each element of the data structure.
- For “Adding data to a particular trial or set of trials”
    - Notice that the rt-task code already does this in the test_stimuli variable. It defines “response” as either “go” or “no-go”.
- You can now skip “Storing data permanently as a file”
- Before completing “Storing data permanently in a MySQL database” we need to get a database set up.

Step 3

Set up MySQL database and tables

First create a MySQL database

To start, go to you phpmyadmin: http://localhost/~<username>/phpmyadmin
(or, if using MAMP, open the start page and select phpMyAdmin)

- It will ask you to Log in: (or not, if using MAMP)
    - Username: If you followed the AMP stack tutorial above, your username should be “root” (no quotation marks!)
    - Password: Your password will be whatever you set it to during the AMP stack tutorial above (You changed it from the default password during the tutorial. Use this new password here.)

- Once logged in, look to the top left corner of the page. You should see “phpMyAdmin” and a row of small buttons below. Under all of this, you should see a skeleton layout with “New” and “information_schema”, among other things. Click “New” to create a new database.
- Under where it says “Create database,” type a name for the database. I’m naming mine manual_database.
- Next, click the “Collation” dropdown menu to the right and scroll down to where it says “utf8_general_ci”. Select that one.

- Then click the “Create” button to create your database.

Next create a table in your new database.

- Now that you’ve created your database, phpmyadmin is asking for you to create a table.
- You don’t have to create a table now (i.e. you could leave it empty and click out of the page and your database would still be there), but we’ll create one so you can hook the table up with jsPsych.
- Where it says “Name” type in a name for your table. I’m naming mine manual_table1 and changing the number of columns to 2.
- Then press go.
- Now it wants you to specify what type of information the 2 columns you just created in your table will hold. I’ve put this information in the next step below.

Specify columns in your MySQL table.

- Each row (currently 2 rows) represents a column in your table.
- We’ll come back to this later, but for now it’s important to note that each key in the data output you saw at the end of running your jsPsych experiment file (i.e. the output that showed “subject” and “condition” keys in each element after adding them in the “Adding data to all trials”) needs a place to go in the MySQL table. Each key, such as “subject”, “condition”, “rt”, among others, must have it’s own column in the table. And the column must be formatted to accept the kind of data you’re sending (i.e. “rt” is sending an integer and “condition” is sending text). When you create columns in the table (here you’re creating just 2 columns), you must specify this information exactly. More on this later.
- Specifying the first column (i.e. the top row)
    - Under “Name” give your column a name. Let’s choose “rt” (no quotation marks) to match the “rt” key in the data output.
    - Under “Type”, choose “INT” (this is probably much too large, but oh well for now.)
    - Leave the rest of the options as is.
- Specifying the second column (i.e. the second, bottom row)
    - Under “Name”, choose “key_press” (again no quotation marks) to match the “key_press” key in the data outputs.
    - Choose the same “Type” as above, “INT.”
- Press save.
- The screen now shows your table structure.
- To see what’s in your table (nothing right now) click the “Browse” tab at the top of the page.
- This should return an empty result.

Step 4

Hook jsPsych up with your new MySQL table

Continue with jsPsych's Data Storage tutorial: http://docs.jspsych.org/features/data/

- For "Storing data permanently in a MySQL database"
  - Under Step 1: I store the savedata.php file in the same directory as my experiment.html file.
  - Under Step 2: Store this file in the same directory as your savedata.php file.
    - Also note that your username and password in this file need to be the same as those you use to enter your MySQL database (i.e. that you used to sign into phpmyadmin). If following the MySQL setup tutorial above, your username is "root" and your password is whatever you chose earlier.
    - My database name is manual_database
  - Under Step 3:
    - Place the code they give you above the "/* start the experiment */" line in your experiment rt-task file.
    - As they instruct you to:
      - Change "my_experiment_table" to the table name you chose. I chose manual_table1. Keep the quotation marks.
      - Change the url path from 'path/to_php/file.php' to point to your savedata.php file. Mine's in the same directory as my experiment file, so I will simply put 'savedata.php'. Again, keep the single quotes they have included.
      - Then comment out ("//") the opt_data: {key: value} line and the preceding comma.
      - Next (in the tutorials/sample code they tell you to do this…), at the end of your experiment code in jsPsych.init, inside the on_finish function and under the line that says "jsPsych.data.displayData();" add the following two lines of code:
        - var alldata = jsPsych.data.getData();
        - save_data(alldata);
      - This will call a function to save the data structure (now in the alldata variable) in your MySQL table
- Next, reload the experiment file in your browser.

- It should still work all the way through, but if you go back to your phpmyadmin page, select your database and then your table and click "Browse" again, no new data will have been inserted into your table.
- Why?

Step 5

Developer tools (your best friend).

Start using developer tools to debug your program.

- Reopen your experiment file in your browser
  - If you were using Safari, it would good to switch to another browser at this point.
  - I like using Google Chrome for this (or Opera).
  - At this point, I've reopened my experiment file in Chrome.
- Now open developer tools by going to:
  - View → Developer → Developer Tools
  - A bar should appear on the right side of your window.
  - If necessary, drag the bar out toward the middle of your screen until you see Elements, Console, and Sources options (there are others, but this is enough).
  - Select Console if it isn't already selected.
- Now rerun the experiment. (You can just press a button to start it and let it run).
  - Side note: If you want the experiment to take less time for the purposes of this manual, just change the all_trials variable from 10 repeats to 2 repeats.
    - var all_trials = jsPsych.randomization.repeat(test_stimuli, 10);
    - becomes →
    - var all_trials = jsPsych.randomization.repeat(test_stimuli, 2);
  - Once the experiment is done, press a button to get to the data skeleton it prints out at the end.
  - You should see a bunch of text spit out to the right side of the screen (the developer tools bar).
- What is this text?
  - This text corresponds to the attempt to insert data into your MySQL table.
    - It's the data array that contains each trial's data in a separate element / object

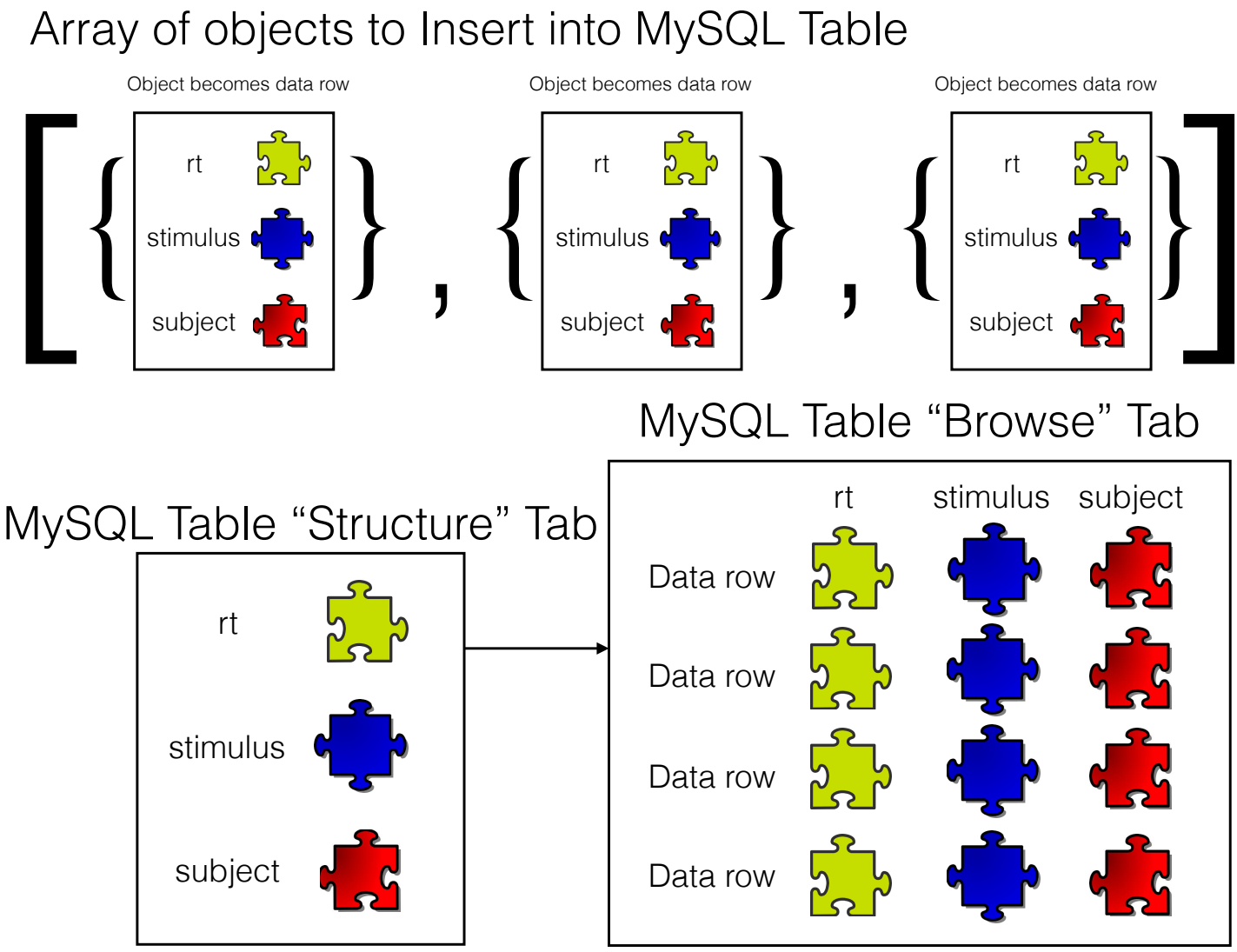
        - Each element in this array begins with [number] => object(stdClass)#number (number) {
        - And each of these object elements will be inserted into your MySQL table in its own row (i.e. each jsPsych trial run has its own row in your table).
    - Note: if you didn't have your program spit out your data to the main screen at the end, i.e. through jsPsych.data.displayData(), the Console would still show your data structure in its attempt to insert into your MySQL table because of "console.log(output)" in the save_data function.

Step 6

Understanding the MySQL table and data insertions.

At this point, it is necessary to understand the MySQL table structure and how objects (each a trial of your experiment) are inserted into tables.

I made a few schematics on the following 5 pages that I hope will be clarifying.



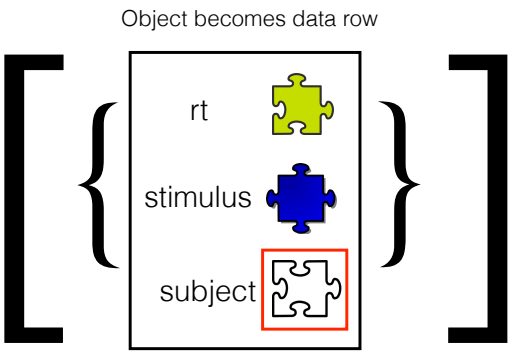
Explanation of the above image:

- Array of objects to Insert into MySQL Table
  - Each of the "objects" inside the curly brackets corresponds to one of the trials / objects in the output you saw after running the experiment (on the screen and in the developer tool's Console).
  - The object has been simplified to include only three "keys": rt, stimulus, and subject (i.e. leaving out things like key_press, response, trial_type, trial_index, etc… that appear in the data output / Console).
  - As it says above each object, each of these objects will become a row in the MySQL table.
- MySQL Table "Structure" Tab
  - This corresponds to the "Structure" tab you encountered in phpmyadmin.
  - Reminder: you saw this tab after specifying names/values for rt and key_press in your table when you created it

- To get back to this page, you can go to your phpmyadmin, select your database, select your table, and go to the “Structure” tab at the top of the page.
- In the image above, rt, stimulus, and subject represent different rows in the Name column in the structure tab of your table in phpmyadmin
- The puzzle pieces represent various specifications you gave the row (i.e. the Type, Attributes, NULL, Default, etc.. which are various columns in the Structure tab of phpmyadmin).

- MySQL Table “Browse” Tab
  - This image corresponds to the table’s “Browse” tab in phpmyadmin.
  - Right now, your table doesn’t have anything in it, but when it does, it will look something like the image above, relative to the Structure image above.

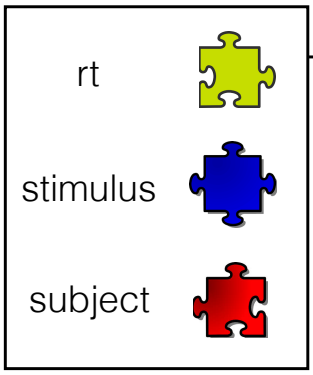
Object to Insert into MySQL Table



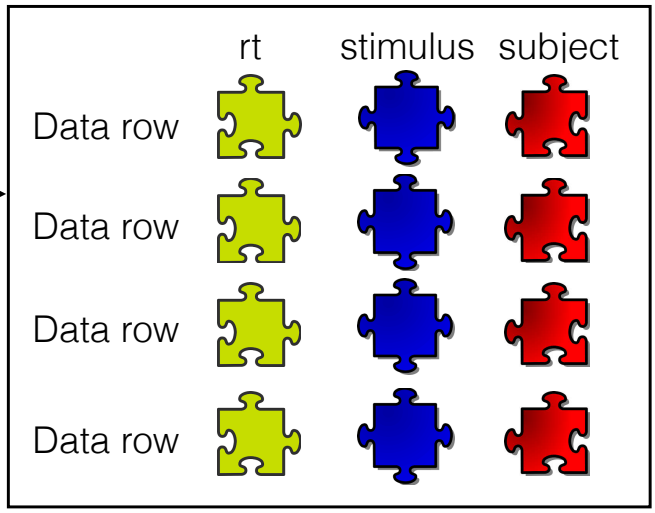
This object will not be inserted into the table.

Why: The subject data sent does not correspond to the subject data the table is expecting.

MySQL Table "Structure" Tab



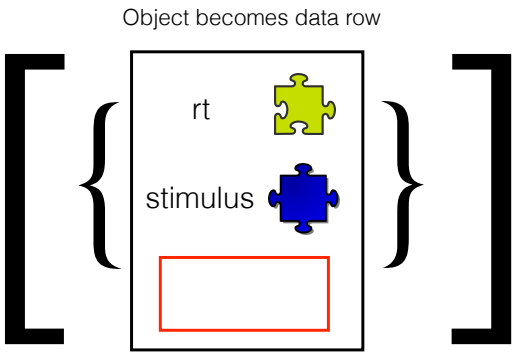
MySQL Table "Browse" Tab



Explanation of the above image:

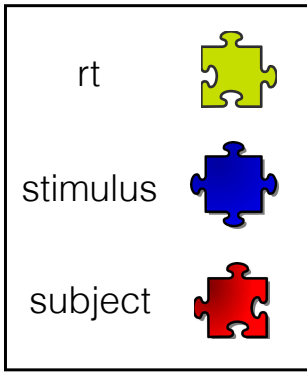
- For the sake of simplicity, we're just working with a single object to insert.
- In this example, no data will be inserted into the MySQL table (not even the rt and stimulus data) because the entire object is inserted at once and the subject data sent (white puzzle piece) doesn't correspond to what the table is expecting (red puzzle piece).
    - An example of this that was discussed in CSE330 might be something like sending "subject" as an integer value when you set the "subject" table Type value to text.
        - i.e. "subject": 9 vs. "subject": "nine" or "9" ← not the same thing and your MySQL table won't take it.
    - Another example could be sending "subject" as a varchar with a length of 300 when you set your limit to a length of 200. MySQL will reject this.
- Bottom line: Make sure the data types/lengths/etc that you're sending correspond to what you specified in your table

Object to Insert into MySQL Table

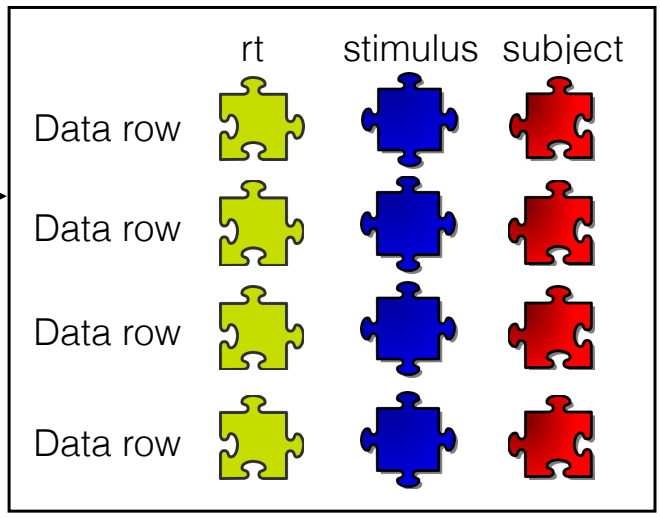


This object will not be inserted into the table.
Why: The table is expecting an rt, stimulus, and subject. No subject was included

MySQL Table "Structure" Tab



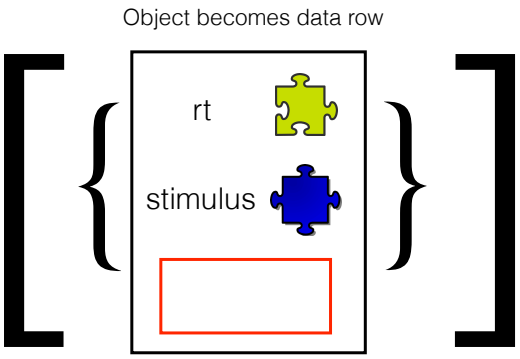
MySQL Table "Browse" Tab



Explanation of the above image:

- In this example, again no data will be stored in your table because your table is expecting an rt, stimulus, and subject, and only received two of those things.
- So then what do you do if not every object (trial) needs to save the same data (say, the subject number isn't recorded on instructions trials but you want to store other information like rt and stimulus from the instructions trials)?
- See next example for the solution…

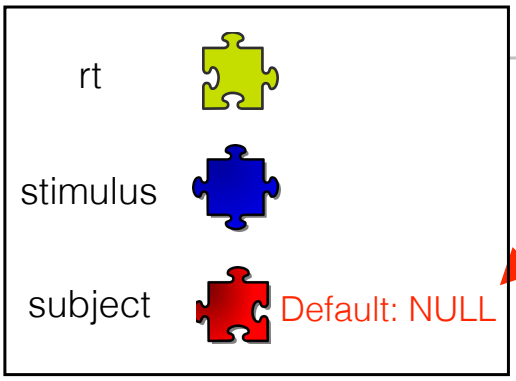
Object to Insert into MySQL Table



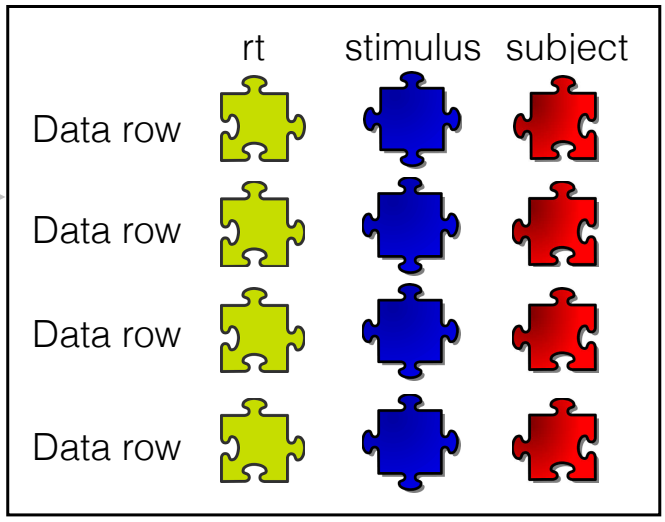
So what do you do if some of your trials (objects) don't include a subject number?

Answer: Tell the subject column in your MySQL table that it's ok to be given a null result.

MySQL Table "Structure" Tab



MySQL Table "Browse" Tab



Explanation of the above image:

- To solve this problem, simply tell your table ahead of time that sometimes subject wont be delivered
  - To do this, go the phpmyadmin "Structure" tab for your table
  - Select "Change" on the row in which you want to add this property
  - Then check the box under "Null"
- In the example above, you can now send an object that doesn't have a "subject" specified and your MySQL table will accept it
- Note: You still must have "rt" and "stimulus" specified. The "Null" specification only works for the variable you give it to. Simply repeat this process for any variables you won't always be sending to the table
- Another note: there is no way to make up for sending too much data to your table. I.e. say you sent "rt", "stimulus", and "LuckyNumber" data. Your MySQL

table doesn't have any column named LuckyNumber so the data won't be inserted. Unlike sending not enough data, which can be addressed by setting a certain column's default value to NULL, there is no way to fix this other than to not send LuckyNumber or to add it as a column in your table.

Step 7

Actually getting your data into your MySQL table.

Hopefully now it's clear why your MySQL table is completely empty when you click the "Browse" button to see your data. None of the objects you submitted exactly matched your table specifications. Let's change that now.

Add new rows to your table structure (i.e. new columns to your actual table)

- In phpmadmin, go to your table and click the "Structure" tab
- Below the two rows, you should see something that says "Add" and a number 1
    - This specifies that you want to add 1 column
    - You can specify where the column should go, but for now, after key_press is fine.
    - Press go
- It will take you to a page like the one you saw before when specifying the rt and key_press rows.
    - Under Name put "trial_type" (no quotation marks)
    - Under Type select varchar
    - Under length/Values put 200 (this sets a maximum varchar length)
    - Leave everything else as is
    - Press save

For the sake of example, rerun your experiment (hopefully you switched the stimuli from 10 to 2 repeats…Consider switching it now if you didn't when mentioned before).

- Once it's finished and you press a keyboard button to get to the data output page, look to your developer tools Console
- Scroll to the bottom of the Console output if necessary
- It should say: Invalid query: Unknown column 'trial_index' in 'field list'
- This is one example of many in which Developer Tools has your back. It's displaying the insertion attempt and MySQL's feedback that literally tells you what your problem is and why the insertion into the table isn't working. Your table doesn't have a "trial_index" column.

- In order to solve this problem, you would check the objects array in the Console to see what kind of data the trial_index column is so that you can properly specify it in your MySQL table. (It's an int).
- To add trial_index as a column in your table, you would do the same thing you did to add trial_type
- However, adding one column at a time can be time consuming, especially if you run the experiment each time to check the console to see why the insertion failed (it will fail at least 5 more times—until each part of each object has a place to go in the table) each time you add a new column. (You could also add all of the rest of the columns at once by looking at the data objects output to the console and specifying each part of the object [key] in phpmyadmin as a new row in the "Structure" tab.)
- If you already have a table set up (say you set one up and want to make a similar one again or send the set up info to a friend, etc…), you can create/ edit/ delete/ etc tables and databases in the "SQL" tab much more quickly using command line.
- So…

Using the command line in phpmyadmin to create a table.

- First, because "condition" is a MySQL reserved keyword, change this key specification in your experiment file code to "conditions" (plural).
    - i.e. condition: condition_assignment → conditions: condition_assignment
- Here is the code you need to create the entire table in one go.
    - ```
      CREATE TABLE `manual_table2` (
       `rt` int(11) NOT NULL,
       `key_press` smallint(11) NOT NULL,
       `trial_type` varchar(200) NOT NULL,
       `trial_index` smallint(6) NOT NULL,
       `time_elapsed` int(11) NOT NULL,
       `internal_node_id` varchar(300) NOT NULL,
       `subject` int(11) NOT NULL,
       `conditions` varchar(200) NOT NULL,
       `stimulus` varchar(200) DEFAULT NULL,
       `response` varchar(100) DEFAULT NULL,
       `correct` varchar(100) DEFAULT NULL
      ) ENGINE=InnoDB DEFAULT CHARSET=utf8
      ```
- To get this going, go to the left side of the phpmyadmin screen, where it shows your databases and tables. Click on your database. (Mine's called manual_database).
- You should see your current table as a row on the page now.
- Next, click the "SQL" tab
      ```

- Copy and paste the above code into the input box on the SQL page. The table above is called manual_table2. You can change the name to whatever you want by replacing the text that says the name with whatever you want.
- Press Go.
- A green check should appear next to a message and on the left side of the screen where you see the skeleton layout that says your database and tables, you should see a new table next to the table you had before. Click on the new table (again, mine is called manual_table2).
- Now go to this table's "Structure" tab to see the structure

Before rerunning your experiment file, make sure to change the table name specification in the experiment file's save_data function to the new table's name.

- var data_table = "manual_table1"; → var data_table = "manual_table2";

Now reload the experiment in your browser.

- When the experiment ends, click a keyboard button to get to the final data output screen.
- Scroll to the bottom of the Console and you should see "successful insert!"
- Now check to see that the data is, in fact, in your table by clicking the "Browse" tab in phpmyadmin (make sure you've selected the right table first!)
- You should see 11 columns. Each row shows a single trial / object.

Step 8

Other things that might be helpful.

More about Developer Tools:

When to use developer tools: Always! Always have that turned on when you're testing your experiment files. In javascript (in the <script> tags in the experiment.html file) you can display things to the developer tools Console by adding "console.log(<stuff to display>)" (No quotation marks). This is really helpful for debugging.

For example, I often add console.log(<variableName>) to see the value of a variable when I'm unsure what's going wrong in my code. Or put console.log(i) inside a for loop to see how many times a for loop with an incrementer "i" executes. Or console.log("hello from inside function B") inside a function to see whether the function is ever called.

In addition, errors will often be automatically displayed in the console. Sometimes the red error text is useless in figuring out what's going wrong, but often times the error messages displayed in the console tell you exactly what you need to change (also if you're unsure what an error message is trying to say, enter the message on Google and without fail someone has had this problem and figured out the solution).

TLDR: Without developer tools and the console, you have very little indication of why something isn't working as it should. You may literally have a blank screen. To save yourself a lot of headaches (and heartache), use developer tools, the Console, and console.log() to debug your code.

MySQL: Get the code you need to recreate a table you already have:
Why this is useful: You've set up table A to work with Experiment X. Now you want to create Experiment Y and you've changed a couple of the data outputs. You want a table similar to table A but with a few modifications. You could make a new table, specifying each column by hand, or…You could generate the code used to create table A and edit it to suite your needs and execute it to create table B to be used for Experiment Y (much faster—and with less room for error if you know your table A is correct).

To generate the code used to create table A: Go to your phpmyadmin. Select the database that contains the table A. Select the "SQL" tab. Enter "SHOW CREATE TABLE <table name> and then press go. A result should appear. Click the "+Options" button just above the Table/Create Table result. Select "Full texts" (instead of the currently selected "Partial texts") and press Go. Now the code needed to generate your table will appear and you can simply copy/paste it into the box in the "SQL" tab to make a new table. Just be sure to have your current database selected and to change the table name and any data specifications you want to be different (you can also manually make changes to the table in the Structure tab after the table is created here).

MySQL: Outputting the contents of your table to an excel file:
In phpmyadmin, select the table you want to output. Go to the "Export" tab at the top of the page. Select "Custom – display all possible options". Under "Format:" select CSV (not CSV for MS Excel). Then scroll to the bottom of the page and check the box next to "Put columns names in the first row". Then press Go. Your file will download and when you open it, it should look just like your table under the "Browse" tab (with a new cell for each option and headers at the top of the page).

jsPsych: Save data after each trial and not all at once at the end:

This is addressed in the jsPsych documentation / forum / example code at different points. I've put a little how-to here:

- Add an "on_finish" function parameter to any trials you want to save individually, like below:

```
/* define welcome message block */
 var welcome_block = {
   type: "text",
   text: "Welcome to the experiment. Press any key to begin.", // Don't forget the ","
   on_finish:  function(){
     var current_node_id = jsPsych.currentTimelineNodeID();
     var currentTrialData = jsPsych.data.getDataByTimelineNode(current_node_id);
     save_data(currentTrialData);
   }
 };
```

- There are other ways to get the current data, but this is the one I choose so that the format of the currentTrialData variable works for what I want to do.
- Also, you can comment out the var alldata = jsPsych.data.getData(); save_data(alldata) in jsPsych.init in your experiment file.
- See an example file in the manual folder under the name "experiment2_withDataStorageEveryTrial.html" that saves data after every trial, rather than at once at the end.

References

jsPsych (of course!): http://www.jspsych.org

- Documentation: http://docs.jspsych.org
- Help Forum (use this!): https://groups.google.com/forum/#!forum/jspsych

Coolest Guides on the Planet (Neil Gee): https://coolestguidesontheplanet.com

- AMP stack / local web development tutorial: https://coolestguidesontheplanet.com/get-apache-mysql-php-and-phpmyadmin-working-on-osx-10-11-el-capitan/

CSE330 – Rapid Prototype Development and Creative Programming - WUSTL: http://classes.engineering.wustl.edu/cse330/index.php/CSE_330_Online_Textbook_-_Table_of_Contents

- In particular, see Module 3:

    - Intro to MySQL: http://classes.engineering.wustl.edu/cse330/index.php/Introduction_to_MySQL
    - MySQL Schema and State: http://classes.engineering.wustl.edu/cse330/index.php/MySQL_Schema_and_State
    - Module 3 assignment: http://classes.engineering.wustl.edu/cse330/index.php/Module_3
- As noted earlier, CSE330's online (wiki) textbook is fantastic for many topics. I refer to it often and much of this manual stems from things I learned in this class/on the class's wiki pages.
- The following provides their Content License information:
    - "The content of this wiki is copyright (c) 2013, Washington University in St. Louis, Department of Computer Science. All rights reserved."