**Online unity demo – running the experiment**

**1) Amazon AWS – DynamoDB set up**

* Create an amazon AWS account: <https://aws.amazon.com/>
* Navigate to “Console” and type “DynamoDB” into the search bar
* Make sure the region you wish to host in is selected top right
* You’ll need to create three new tables:
  + First has a table name “schoolUserData” with a primary key “tokenId”(string)
  + Second has a table name “schoolTrialData” with a primary key “tokenId”(string) and sort key “trial”(string)
  + Third has a table name “tokenTable” with a primary key “tokenId”(string)
* Return to the console and type “IAM” in the search bar
* Click “Policies” -> “Create Policy”
* Select “DynamoDB” as the service
* Under “Actions”, check the following boxes (5 Total):
  + Dropdown menu “Read”:
    - “GetItem”
    - “Query”
  + Dropdown menu “Write”:
    - “PutItem”
    - “UpdateItem”
    - “UpdateTable”
* Under “Resources”, select “all resources”
* Click “Review policy” and give it a name like “ParticipantAccess”
* Navigate back to IAM and select “users”
* Click “Add user”, give it a username like “Participant”, check the “Programmatic access” box
* Click “Next: Permissions” and “Attach existing policies directly”
* Then type in the name of the policy you created earlier and check the box next to it
* Click “Next” through to the end until you’re given an ACCESS KEY and SECRET KEY for the user
* Copy and paste your ACCESS KEY and SECRET KEY into a .txt file, this is the only time you’ll get to view the secret key so make sure you have it right

**2) Uploading your generated tokens to DynamoDB**

* Install node.js: <https://nodejs.org/en/>
* Open the command prompt and type “npm install aws-sdk”
* Open “genericTokenGenerator.js” from the root directory in a code editor that will let you run code snippets through node (Visual Studio code will do the trick)
* Copy and paste your ACCESS KEY and SECRET KEY from earlier intothe generateTokens() function. You may also change the server “region” and “endpoint” settings if you’re not hosting on us-east-2 (my default).
* Input the number of tokens you require in the generateTokens() function at the bottom
* Ctrl+alt+n will allow you to run the code
* Open your browser and check the “tokenTable” table in DynamoDB to confirm all tokens have been updated

**3) Unity project**

* Download the unity project folder onto your computer
* Navigate to Assets -> Plugins -> jsplugin.jslib and open in your code editor
* Copy and paste your ACCESS KEY and SECRET KEY from earlier into **all four** functions. You may also change the server “region” and “endpoint” settings if you’re not hosting on us-east-2 (my default).
* Now open the project using unity ver. 2019.4.4f1
* File -> build settings -> select WebGL
* Make sure both scenes are selected and then click build
* Navigate to index.html in the folder you’ve just built to and open in a code editor
* Insert the following line in the <head> of your .html (at the top of the <script src=> stack):
  + <script src="https://sdk.amazonaws.com/js/aws-sdk-2.7.16.min.js"></script>

This will allow you to interface with the aws SDK using javascript

* Now go back to the parent folder of index.html and archive it along with the “Build” and “TemplateData” folders into a .zip file. **NOTE: this does not require you to zip the parent folder of your project, rather, you archive the separate constituents together within your parent folder**

**4) Amazon AWS – static hosting**

* Navigate to the AWS console
* Type “Amplify” into the search bar
* “Deploy” an app -> deploy without Git provider
* Give your app a name and an environment name (neither matter)
* Click drag and drop and drag your.zip folder in -> save and deploy
* Once it has finished creating your app you’ll be able to follow a link to your unity game
* Test it out and check the data is being saved to the database correctly. You should have a single entry in the userData table for each user, and 10 entries in the trialData table for each user (one for each trial)

**5) Downloading data**

* Make sure you have node.js installed (as per 2.)
* Download (or clone) dynamodbtocsv from: https://github.com/edasque/DynamoDBtoCSV/
* Open command prompt, navigate to folder and run “npm install”
* Navigate to the folder and edit the config.json to include your AWS credentials
* To download a table to csv you can use the following line:
  + node dynamoDBtoCSV.js -t YOUR\_TABLE\_NAME -f FILENAME.csv -v{}
* Note: you’ll get errors if you don’t use the –v{} argument (eventhough this isn’t stated in the readme!)

*NOTES ABOUT IMPLEMENTATION*

So the way this works is by interfacing with the AWS SDK for DynamoDB using javascript. You can’t normally execute javascript in Unity so we’ve had to write a plugin (“jsplugin.jslib”) which defines the functions we’d like to call in javascript. We can then import these functions from Unity classes by [DllImport(“\_\_Internal”)] using System.InteropServices. Then at any point in our script we can call them as we would any other function.

Another feature of this is the use of singletons to pass userdata between scenes. This is handled by the UserInfo.cs script attatched to an empty object (SaveData) in the first scene. We can then call any of these parameters in future scenes by use of UserInfo.Instance.yourparameterhere. This is especially useful when we’re working with two separate tables that we we’ll want to link on processing.

**Online experiment ICON protocol**

Experiments will be run with samples from three broad categories: MTurk; Schools; and Web. The requirements below apply equally to all three types of experiment. Subsequently, a table is presented in which specific recommendations are given for each individual category.

**1) Database convention**

Each experiment will have a minimum of two tables:

1) UserData – demographics and consent timestamp

2) TrialData – trial by trial data

A third table will be necessary is necessary if you are running an experiment which you wish to restrict access to (i.e. schools)

3) TokenTable - unique anonymous tokens

The lab uses a communal AWS account and many tables will accrue over time. Accordingly, please adopt the following naming convention to assist with administration:

[ RESEARCHER’S INITIALS]\_[EXPERIMENT]\_[VERSION]\_[PARTICIPANT POOL]\_[TABLE\_NAME]

For example, researcher *Joe Bloggs* is *piloting* their experiment, *Cannon Chaos* using Mturk participants. Their tables, have the following names:

JB\_CC\_PILOT\_MTURK\_UserData

JB\_CC\_PILOT\_MTURK\_TrialData

**2) Demographic data**

The following fields should be collected for all participants:

TokenId (str) – anonymous token assigned by experimenter

Handedness (str) – “Left” or “Right”

Gender (str) – “Male” or “Female”

Input device (str) – “Mouse/Trackpad/etc” (if necessary)

Browser version (str) – the browser and version being used

HeightPx (dbl) – window height in pixels

WidthPx (dbl) – window width in pixels

PxRatio – level of zoom user is using (actual screen dimensions calculated by multiplying by ratio)

The TokenId assigned will depend on the category of experiment (see table). Handedness and gender are reported by the user. Browser version and screen resolutions are collected automatically by functions assigned to the javascript plugin:

getScreenWidth()

getScreenHeight()

getPixelRatio()

getBrowserVersion()

These should be called from the first scene in your experiment and the return values sent to the UserData table.

Please note, additional data should only be collected in accordance to the table for each category. An important principle to follow is that we do not store identifiable information on AWS servers.

**3) Consent**

You should not send any data to AWS until participants have provided informed consent. All consent forms should include a list of affirmative statements which participants must tick to indicate they meet the inclusion criteria. The content of the consent form will depend largely on the category of experiment you are running. Examples for each are given in the appendix as a starting point. Please note, this should be done in engine for *School* and *Web* experiments, but via Qualtrics for *MTurk* (see table)

**4) Full screen and Cursor Locking**

It is important to ensure that participants are not distracted while performing our tasks and the experiment should only progress while in fullscreen mode. Fullscreen can be toggled in Unity:

Screen.fullScreen = true/false; (ON/OFF)

Moreover, the cursor should be hidden from participants so only the stimuli on screen are visible. This can be toggled in Unity:

Cursor.lockState = CursorLockMode.locked; (ON)

Cursor.lockState = CursorLockMode.none; (OFF)

If the participant exits either mode (by default browsers do this when they press escape), the experiment should be suspended. This can be achieved by adding javascript event listeners in the jslib.js plugin which call a pause function in Unity when they trigger:

fullScreenListener()

lockListener()

Note that if you want the mouse to control stimuli on screen, you will no longer be able to do this by tracking the pointer but will need to use the raw input from the mouse.

**5) Ensuring the view is consistent across different aspect ratios (2D)**

Participants using native resolutions that differ from the development machine may have elements of the screen cut off. To ensure consistency across non-standard resolutions, it’s important to change the camera settings in these cases. Attach the *CameraController*.*cs* script to the camera in your scene and call these functions from Unity when the fullScreenListener() event fires. Please note that the *CameraController.cs* script is targeting a native 16:9 aspect ratio and you should change this variable (line 18) if you are targeting a different default aspect ratio.

**6) Pausing and exiting the experiment**

To meet the ethical requirements set out by the BPS, participants must be able to pause or withdraw from the experiment at any time. At all times during your experiment the following text should be clearly visible in the corner of the screen:

“Press **ESCAPE** to exit full screen and pause”

The following text should then be present on the pause screen:

“Paused. Click on the full screen icon in the bottom right hand corner of the window when you are ready to resume the experiment. If you would like to withdraw from the study, you are free to close the browser window at any time”

To prevent spurious clicks from closing the browser window accidentally, it’s possible to trigger a warning window when users attempt to close or refresh the window. This is achieved using a js function from the jslib.js plugin:

UnloadListener()

Please note that the latest browsers no longer support the use of custom text in this window and it will always state the same warning. It is also recommended that you timestamp and log any pauses to identify those who may have taken a disproportionate number or length of breaks.

**7) Handling withdrawals**

All users with incomplete data should be treated as having withdrawn. Accordingly, before downloading data from AWS, filter the *TrialData* table for those users who do not have *n* rows equal to the number of trials in your experiment and delete them from all tables.

**SPECIFIC CONSIDERATIONS FOR EACH CATEGORY**

|  |  |  |  |
| --- | --- | --- | --- |
|  | **MTURK** | **WEB** | **SCHOOLS** |
| **TOKENIDs** | Passed from MTurk (via Qualtrics) as their worker ID. This allows you to confirm workers have completed the task for payment. | Randomly generated, non-editable field on form. Recommended to use [adjective]-[colour]-[animal] using Em’s dictionaries | Randomly generated [adjective]-[colour]-[animal]. These are generated using the tokengenerator.js script and should be passed to teaching staff prior to commencement so they can link it to individual pupils. Pupils then input the key they’ve been given which is subsequently linked to their school data offline using the key provided by staff. |
| **DEMOGRAPHICS** | Collected using Qualtrics before redirecting to experiment. This permits us to collect additional demographic information (e.g. questionnaire measures linked to our data) and ensure we only accept unique IPs | Form programmed in Unity engine. May additionally request geographic information at the “nationality” level. | Form programmed in Unity engine |
| **CONSENT** | Displayed in Qualtrics before redirected to experiment.  Must make explicit the requirements for payment to be made (e.g. follow instructions carefully and complete all trials) | Displayed in Unity engine on initial scene.  May also collect age in years.  May also provide an additional tick box which allows participants to participate in the experiment without any data being collected. This should be selected by default if participants are younger than 18. | Displayed in Unity on initial scene.  May also provide an additional tick box which allows participants to participate in the experiment without any data being collected. This should be flagged up by teaching staff who have received opt-out consent forms from parents prior to the experimental session. |
| **EXITING** | Participants should be warned in the pause screen that they will not be paid if they exit the experiment prematurely. |  | Progress should be saved temporarily to allow participants to rejoin and resume with their progress if there are any shenanigans in the classroom.  This can be achieved by including experimental and trial progress fields in the token table and updating it with these parameters when the UnloadLIstener is called. Subsequently, on experiment start, the tokenTable can be checked for progress before loading the scene. |

**APPENDIX – INFORMATION SHEET/CONSENT FORM TEMPLATES**

**MTURK**

**WEB**

**SCHOOL**

**Participant Information Sheet**

**Born in Bradford: Online Pilot Scheme**

**Young People’s Movement and Coordination Study**

**Why are the Born in Bradford team looking at young people’s movement and coordination?**

The Born in Bradford team and the University of Leeds want to understand more about the development of hand-eye coordination and how it relates to educational outcomes (e.g. scores in maths tests).

**What will the Born in Bradford team do?**

They will:

1. Provide you with a key to play a video game for about 20 minutes at the start of class

2. Access data held by the school on your progress in reading, writing, and maths.

**Do I have to take part in this project?**

No, you don’t! Your parents have got their own letter explaining this project and if they are happy for you to take part they don’t need to do anything.

You will only be able to participate in the study if you click “accept” below. If you later decide to withdraw, please contact [xxxx@leeds.ac.uk](mailto:xxxx@leeds.ac.uk) to be removed from the study.

**What are the benefits of taking part?**

1. You get to play a game and participate in one of the largest studies of its kind in the world.
2. Future research will use our findings to help identify teenagers who need extra help to get good results.
3. You will also be helping to improve how computers learn to perform similar tasks.

**What will we do with your data?**

All data will be stored securely by the Born in Bradford team and the University of Leeds. Your data will be given a number, not a name, so you won’t be identifiable. We plan on using this data in scientific publications and your name will not be included. We will only share our assessment data with your school, who may use it to help you in future. No other organization will have access.

**Have you got any questions about this?**

If you have questions you can ask your Form Tutor or email us at [xxxx@leeds.ac.uk](mailto:psjpp@leeds.ac.uk)

**Student Consent Form**

I consent to participating in the Born in Bradford (BiB): Secondary School Pilot Scheme, Coordination and Movement Assessment of 6-16 year olds.

I have discussed this with a teacher and:

* I confirm that I have read and understand the Participant Information Sheet
* I have had the opportunity to ask questions and had them answered
* I agree that data gathered in this study may be stored anonymously and securely, and may be used for future research
* I understand that my participation is voluntary and that I am free to withdraw at any time without giving a reason.

Please tick each box and click “Accept” below to agree to these terms. If you do not, click “Reject” and exit your browser. Make a note of the secret key given to you by your teacher. If you have any questions in future regarding this research, send this in an email to us at [xxxx@leeds.ac.uk](mailto:psjpp@leeds.ac.uk)