Mactivision Testable Artifact

Instructions

If you have any problems setting up the testable artifact please contact *Bryan Chiu* (chiub1@mcmaster.ca) on teams.

Please report all issues/bugs found during testing of this artifact to the issue tracker in our repo at: https://github.com/BryanChiu/Mactivision/issues

Project Introduction

Our group has developed various cognitive and motor abilities disguised as mini-games. Each test is a play-through of a series of mini-games, called a *Mini-game Battery*. After each mini-game is completed, the user's gameplay is logged and sent to a server to be stored. Player data will be processed by researchers to measure the user's performance of the cognitive and motor abilities used to complete the challenges of each mini-game. Each Battery can be tuned by researcher's to measure the changes in a user's performance when using different mini-game parameters.

This artifact contains a WebGL build of our Unity battery and mini-game code, coupled with two python servers, one to host the client interface (Mini-games), and the other to host the server which the client sends and receives battery data from.

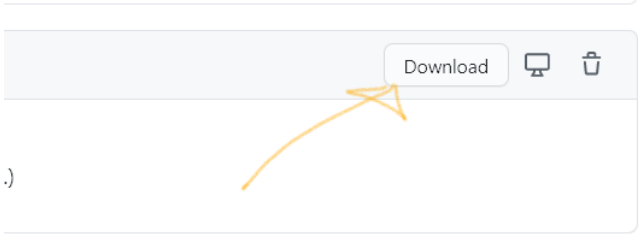
Tools

The client and server python scripts run using Python 3. Python 2 will not work because we use specific modules named differently. There are instructions on how to install Python 3 at https://www.python.org/downloads/

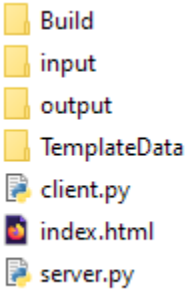
For browsers we recommend chrome or firefox.

Running the Battery

1. Download link https://github.com/BryanChiu/Mactivision/blob/master/artifact.zip



2. Extract **artifact.zip**
3. You should see the following files:



4. Run client.py from the unzipped artifact directory

```
> python client.py
```

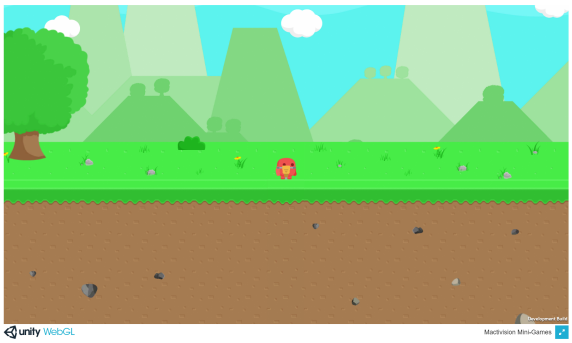
5. Run server.py from the unzipped artifact directory

```
> python server.py ./input/Battery.json
```

6. client.py starts a server on 127.0.0.1:3000
7. server.py starts a server on 127.0.0.1:8000
    a. Server.py might produce errors if you are missing files and folders.
8. Navigate your webgl compatible browser to 127.0.0.1:3000
9. You should see the following:

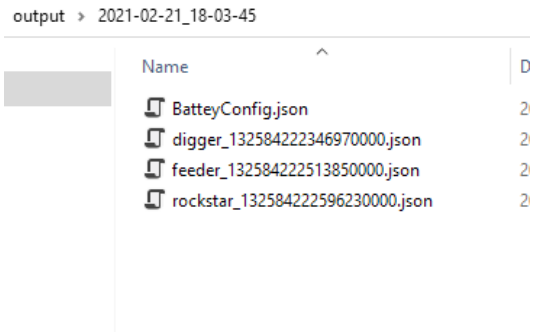
10. The **debug console** on the game exists to output error messages. If the battery loaded correctly the console should not have errors and the start button should be active.

11. Click the start button to begin the battery. Depending on the configuration, you should now see a game loaded.



12. When the battery is complete you should see this screen:

13. After the battery is complete, the output folder will contain player data describing the actions of the player for each game and the configuration used for the battery. The results should look something like this:



14. Feel free to test different configurations and see what output you get.
15. You may want to use something like http://json2table.com/ to make the json more readable.

Editing the Configuration File

Digger

Key	Default	Accepted	Description
"DigAmount"	100	$\mathbb{N} > 0$	Amount of keystrokes to hit the bottom. Will round up to multiples of 10 as there are 10 blocks to dig. A `DigAmount` of 100 means each block will take 10 keystrokes to break
"DigKey"	“B”	Unity KeyCodes	Key used to dig

Feeder

Key	Default	Accepted	Description
"Seed"	null	string in “double quotes”	Seed used for for random-number-generation
"MaxGameTime"	90.0	$\mathbb{R} > 0$	Maximum seconds elapsed before game ends
"MaxFoodDispensed"	20	$\mathbb{N} > 0$	Maximum food items dispensed before game ends
"TotalFoods"	6	$1 \leq \mathbb{N} \leq 7$	Amount of unique food items used
"AverageUpdateFrequency"	3.0	$\mathbb{R} > 0$	Average amount of food items dispensed in between food choice updates. Will always be at least 1, or an update every dispense
"UpdateFreqVariance"	0.3	$0 \leq \mathbb{R} \leq 1$	Variance of `AverageUpdateFrequency`. If `AverageUpdateFrequency` is 3, a

			variance 0 means it will always be 3 dispenses in between food choice updates. A variance of 1 will allow +/- 4 (-1 to 7), so it will be 1 to 7 dispenses between food choice updates

Rockstar

Key	Default	Accepted	Description
"Seed"	null	string in “double quotes”	Seed used for for random-number-generation
“MaxGameTime”	30.0	$\mathbb{R} > 0$	Maximum seconds elapsed before game end
"RockstarChangeFreq"	1.0	$\mathbb{R} > 0$	Average frequency in seconds at which the destination of the rockstar changes
"RockstarVelocity"	2.5	$\mathbb{R} > 0$	Constant velocity of the rockstar
"SpotlightVelocity"	3.0	$\mathbb{R} > 0$	Constant velocity of the spotlight
"MeterChangeFreq"	2.0	$\mathbb{R} > 0$	Average frequency in seconds at which the meter changes velocity
"MeterMinVel"	5.0	$\mathbb{R} > 0$	Minimum amount/second the meter drops. Meter level is in range 0-100
"MeterMaxVel"	30.0	$\mathbb{R} > 0$	Maximum amount/second the meter drops. Meter level is in range 0-100
"MeterUpVel"	60.0	$\mathbb{R} > 0$	Constant amount/second the meter is raised by the player. Meter level is in range 0-100
"LeftKey"	“A”	Unity KeyCodes	Key used to move the spotlight left

"RightKey"	“D”	Unity KeyCodes	Key used to move the spotlight right
"UpKey"	“L”	Unity KeyCodes	Key used to move the meter up