Treasure Hunt

Todo list:

Use transistor to switch on servo only when there’s something to move to avoid jitter.

# What the viewer sees

The machine requires a way to attract the attention of a passer-by, and to engage them into interacting.

A dormant mode might cycle through advertising. If the advertising is itself “out-of-keeping” with the theme of the machine then it requires a kind of “portal border” through which the advert is viewed. Better would be an adapted advert made to look in the Victorian style.

A demo mode might show how the game is played.

Server

Station

SPA

# Requesting New Access Code

SPA calls REST.GenerateNewAccessCode, passing in the GameId.

Server sends ClientMessage, with cmd = “GenerateAccessCode”.

Server return TRUE is code can be generated, else FALSE (if game is busy/offline)

The Game generates the new code and informs the Server via REST.UploadStationAccessCode

The Server informs the Client that the code was generated and the Client shows the input controls

# Sending Command from SPA to Station

Use REST call to Server:

**PassCommandToStation**

Internally calls:

**SignalRClientComms.PublishMessageToSignalRClients**

Which takes arguments that are the members of the ClientMessage class

public class ClientMessage

{

public string Sender;

public string StationId;

public string Direction;

// Token so that the clients know if the message is for them

public string AccessToken;

// The Command

public string Command;

// The parameters

public string Parameters;

}

Can be tested using:

<https://sandgatethapi.azurewebsites.net/api/messagetest>

<http://localhost:7071/api/messagetest>

# UI Design

Each game should have a banner logo in the proportion of 4W:1H. This will appear in several places, both on the UI and Viz in order to tie the two together visually.

Mobile ui has

15% height for banner

80% for ui in total.

5% footer.

# Machine Design

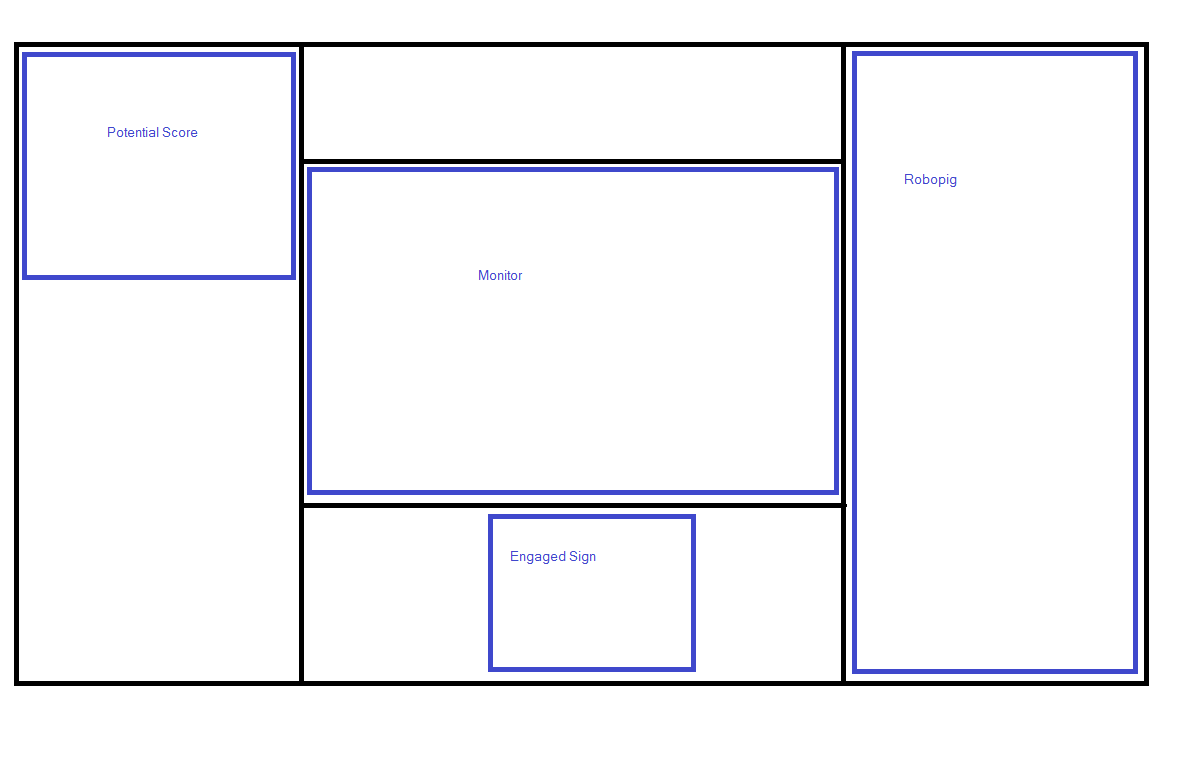
## Central Pi

Runs the game via DotNet core console app. Has Wifi connection.

Runs the visualisation via SPA, and connects to screen display.

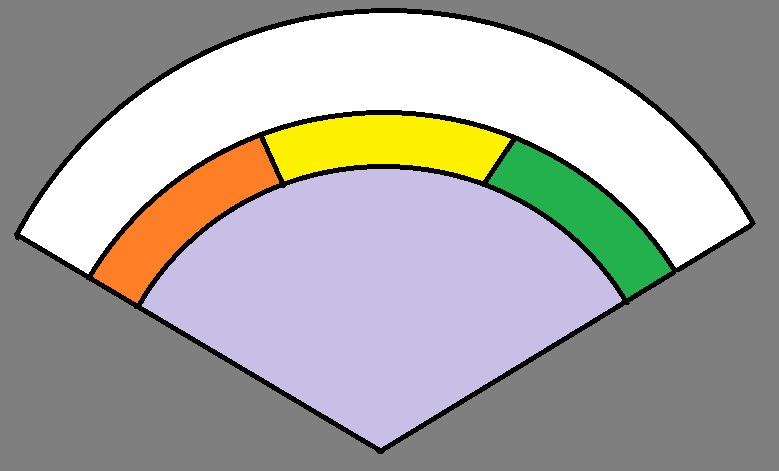
Has direct connection to Motion sensor and Light Sensor and publishes GameState to all peripherals via I2C broadcast.

## Displays



All displays should have a heartbeat LED.

### Potential Score Marker



Servo driving marker

Reacts to potential score only when the game is in playing mode

### RoboPig

**Activated**

Almost nothing.

**OnlineReady**

Sideways glance, little jig now and then

**OnlineDemo**

Glance up to screen, nod, wave

**OnlineDormant**

Snore

**Authenticating**

Charge power banks, pig head bowed, lifeless.

**PreGame**

Means auth complete. Signals start of pignition, the main show.

Blade switch flips, Power bank discharges through light lines, electrodes swing round, then sparks in tubes. Pig shudders to life.

**Playing**

Give feedback regarding correct or incorrect answer

**PostGame**

Dances little jig, then goes into sleep mode. Light lines go in reverse from electrode, or the power starts to get intermittent.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Time/Element** | **Blade switch** | **Powerbank** | **Conduits** | **Electrodes swings** | **Electrodes** | **Pig motors x 5** |
|  | *1 PWM* | *1 DO* | | *2 PWM* | *1 DO* | *5 PWM* |
| 0 | Off | Full/Pulse? | Off/Dim | Away | Off | Dormant |
| 300 | On | Full | Off/Dim | Away | Off | Dormant |
| 1300 | On | Discharging | Increment | Away | Off | Dormant |
| 1500 | On | Discharging | Lit | Move to Touch | Off | Dormant |
| 1700 | On | Discharging | Lit | Touching | Strobe | Shocked |
| 2500 | On | Discharged | Decrementing | Touching | Strobe | Shocked |
| 2500 | On | Discharged | Off/Dim | Move to Away | Off | Shaken |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |

Tttt:[Bla

SetElements(timeMs, bBladePos, PowerbankState, ConduitState, ElectrodeState)

Followed by

SetPigServos(timeMs), pos1, pos2 etc)

ElectrodeState: Away, MoveTo, Strobe(after fixed interval), MoveBack(stop strobing immediately)

### EngagedSign

Servo driving rotating Vacant/Engaged Sign. Vacant means “Online\_xxx”

Led strip for background. Displays colours for

Online\_Demo

Online\_Dormant

Online\_Ready

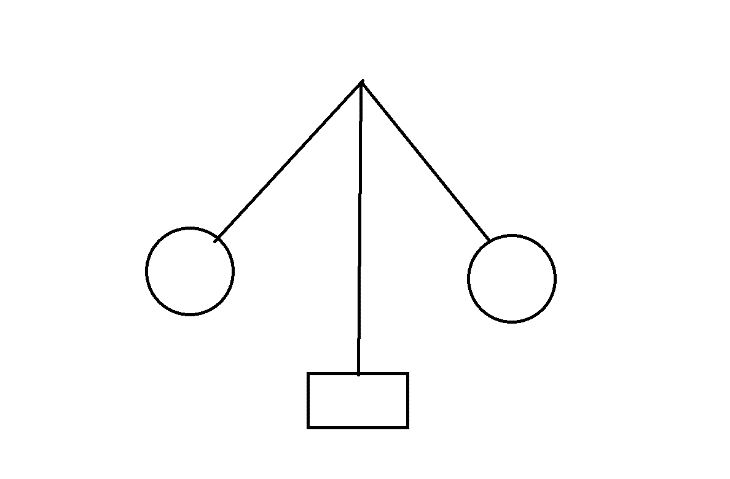
Other

### LED Matrix

Matrix should be used for most states, with animations

|  |  |
| --- | --- |
| **State** | **Action** |
| Online\_Ready | Spark |
| Online\_Dormant | Random Blob |
| Online\_Demo | “Scan Code” |
| Authenticating | Side to side |
| PreGame | “START” |
| GamePlaying - FetchQuestion | SidetoSide red |
| GamePlaying - DisplayQuestion | Chevron to outside @ 1Hz |
| GamePlaying - DisplayAnswer-Correct | Quick green sin wave |
| GamePlaying - DisplayAnswer-Incorrect | Quick red sin wave |
| PostGame | Random Blob |

### Steam Engine Governor



Requires bracing at top and bottom, and a coupling to the motor. And pivots for the arms

### Dashboard

Shows status information to confirm what’s going on. Just for my benefit, but information displayed on dials and lights with fantasy naming.

Light Sensor, boolean

Movement Sensor, boolean

Game State 1-8

Internal State 0-3

Last i2c signal

### Cogs and wheels

Just coz they’re fun.

### Some form of Clock

Mechanical hands aren’t new or impressive, and are hard to calibrate if driven from servos.

Some kind of illuminated display could be good. Backlit digits are simple. Ought to visibly change every second, so lights would be more robust. Viewer may need to puzzle through the analogy. Combine lights with mechanical. A string of lights to represent seconds, with the string filling up over the course of 1 minute

### I2C/Power Distributor

**Input**

5V from high capacity supply

I2C from RasPi

Both using screw connectors

**Output**

5 x 4pin Aviation sockets

VA meter showing power usage. Or have that separate.

# I/O Logic

# Project folder organisation

App

Components (common comps for all games)

Maybe - common-styles.scss (common styles for all games)

Games (for game specific code)

[Game1]

Style.scss

UI

Viz

Pages

Parts (like “Components” for things that only this game has)

Assets

[Game2]

Interceptors

Services

Shared

Views (for non game related views, which will always be related to UI)

## Game State Machine

Deactivated

PostGame

GamePlaying

PreGame

Authenticating

Online

Activated

Initialised

Online\_Dormant, low activity mode, can be woken by QCode or other cue

Online\_Ready, ready for Qcode scan. Default state

Online\_Demo, Performing attractive animation.

Goes into demo when no activity for long period. Come out of demo when demo finished.

Initialised = 0, // Started

Activated, // Registered at server, but not available for clients

Online, // Registered at server, and available for clients

Authenticating, // Access Code generated, waiting for client to send code

PreGame, // |

GamePlaying, // | - Client connected

PostGame, // |

Deactivated //

## Question format

Since each question has a numerical answer, they lend themselves not just to a MoreOrLess scenario, but also a “Higher/Lower” type game. The latter requiring that questions get sorted into categories of answers of approximately the same magnitude.

Suitable categories might be:

Battle Dates

River length

Birth/Death dates of famous people

Country population/land mass

Music pop chart