Appendices

Appendix A : Git Repository
I have been granted my supervisor's approval to provide reference code on my GitHub repository for the project at www.github.com/AMRoche/MakeTheCosm

Appendix B : Cosm Datastream JSON object example


```
{
  "title" : "Cosm Office environment",
  "description" : "test of manual feed snapshotting",
  "feed" : "http://api.cosm.com/v2/feeds/504.json",
  "id" : 504,
  "status" : "frozen",
  "website" : "http://www.haque.co.uk/",
  "updated" : "2010-06-25T11:54:17.463771Z",
  "created" : "2010-03-03T23:43:01.238734Z",
  "version" : "1.0.0",
  "private" : "false",
  "creator" : "https://cosm.com/users/hdr",
  "location":
    {
      "disposition":"fixed",
      "ele":"23.0",
      "name":"office",
      "lat":51.5235375648154,
      "exposure":"indoor",
      "lon":-0.0807666778564453,
      "domain":"physical"
    },
  "tags" : [
    "Tag1",
    "Tag2"
  ],
  "datastreams" : [
    {
      "at" : "2010-06-25T11:54:17.454020Z",
      "current_value" : "999",
      "datapoints" : [
        {
          "at" : "2005-06-25T11:54:17.000000Z",
          "value" : "999"
        },
        {
```

```
          "at" : "2006-06-25T11:54:17.000000Z",
          "value" : "123"
        }
      ],
      "id" : "3",
      "max_value" : "999.0",
      "min_value" : "7.0",
      "unit": {
          "type": "conversionBasedUnits",
          "label": "label",
          "symbol": "symbol"
      },
    },
    {
      "at" : "2010-06-24T10:05:49.000000Z",
      "current_value" : "0000017",
      "datapoints" : [
        {
          "at" : "2010-04-12T11:31:51.133782Z",
          "value" : "999"
        }
      ],
      "id" : "4",
      "max_value" : "19.0",
      "min_value" : "7.0"
    }
  ]
}
```

Appendix C : Highcharts chart creation example

```
       new Highcharts.Chart({
           chart: {
               renderTo: document.getElementById(feedId + dataId +"graph"),
               type : 'area',
               marginRight: 10,
               events: {
                   load: function() {

                       // set up the updating of the chart each second
                       var series = this.series[0];

                   }
```

```
            }
        },
        title: false,

            credits: {
                enabled: false
            },
        xAxis: {
            type: 'datetime',
            tickPixelInterval: 150
        },
        yAxis: {
           max : Math.ceil(maxVal),
           min : Math.ceil(minVal),
            title: false,
            plotLines: [{
                value: 0,
                width: 1,
                color: '#808080'
            }]
        },
        tooltip: {
            formatter: function() {
                    return Highcharts.dateFormat('%Y-%m-%d %H:%M:%S', this.x)
+'<br/>'+Highcharts.numberFormat(this.y, 2);
            }
        },
          plotOptions: {
            area: {
                        threshold : -999999,
                marker: {
                    enabled: false,
                    symbol: 'circle',
                    radius: 2,
                    states: {
                        hover: {
                            enabled: true
                        }
                    }
                }
            }
        },
        legend: {
```

```
            enabled: false
        },
        exporting: {
            enabled: false
        },
        series: [{
            name: 'Streamed Data',
            data: (function() {
                // generate an array of random data
                var data = [],
                    time = (new Date()).getTime(),
                    i;
                for (i = graphUpdateTime * graphResolution * -1; i <= 0; i++) {
                    data.push({
                        x: time + i * 1000,
                        y: -999999
                    });
                }
                return data;
            })()
        }]
    })
```

Appendix D : Chosen data points to sonify
Description of object;

```
{
“FEED_ID_AS_DETERMINED_BY_COSM” : [
        [
                “DATA_POINT_ID”,
                ”CURRENT_VALUE”,
                MAXIMUM_VALUE,
                MINIMUM_VALUE,
                HIGHCHARTS_CHART,
                        [
                                “DISCRIMINITORY_SYMBOL”,
                                THRESHOLD,
                                ”SOUND_STRING”,
                                TO_PLAY
                        ]
                ]
        ]
}
```

Data types;

While data types are, for the most part, immaterial in Javascript, the application expects certain variable types in certain positions.

```
{
“FEED_ID_AS_DETERMINED_BY_COSM” : [
        [
                string,
                string,
                float,
                float,
                Highcharts chart Object,
                        [
                                string,
                                float,
                                string,
                                boolean
                        ]
                ]
        ]
}
```

Object in the wild;

```
{
"123597":[
                ["CO","1012",1683,0, Highcharts Chart,
                        [">>", 20, "2||chimes||Chimes_b", true]
                ],
                [“DTH-H”,”20”,40,0, Highcharts Chart,
                        ["|==|", 10, "2||chimes||Chimes_b", true]
                ]
        ]
}
```

Appendix E : Audio format dependencies per browser
sourced from : http://www.w3schools.com/html/html5_audio.asp

Internet Explorer 9+	YES	NO	NO
Chrome 6+	YES	YES	YES
Firefox 3.6+	NO	YES	YES
Safari 5+	YES	YES	NO
Opera 10+	NO	YES	YES

Appendix F : Audio data JSON structure

```
{
	"opts": [["mp3", "mpeg"],["wav", "wav"]
	],
	"init": ["0||samples||test_sample", "0||chimes||Chimes_a", "2||chimes||Chimes_b",
"4||chimes||Chimes_c", "6||chimes||Chimes_d", "7||chimes||Chimes_e", "9||chimes||Chimes_f",
"11||chimes||Chimes_g"],
	"samples": [
		["test_sample", "A sad, sad music box."],
		["Snare_1", "A snare drum."],
		["Snare_2", "Another snare drum."],
		["Bass_1", "A bass drum."],
		["Bass_2", "Another bass drum."],
		["High_Hat_1", "A high hat."],
		["Ride_1", "A ride cymbal."]
	],
	"guitar": [
		["Guitar_a", "A Telecaster Guitar playing the note of A."],
		["Guitar_ab", "A Telecaster Guitar playing the note of A Flat."],
		["Guitar_b", "A Telecaster Guitar playing the note of B."],
		["Guitar_c", "A Telecaster Guitar playing the note of C."],
		["Guitar_d", "A Telecaster Guitar playing the note of D."],
		["Guitar_db", "A Telecaster Guitar playing the note of D Flat."],
		["Guitar_e", "A Telecaster Guitar playing the note of E."],
		["Guitar_eb", "A Telecaster Guitar playing the note of E Flat."],
		["Guitar_f", "A Telecaster Guitar playing the note of F."],
		["Guitar_g", "A Telecaster Guitar playing the note of G."],
		["Guitar_gb", "A Telecaster Guitar playing the note of G Flat."]
	],
	"chimes": [
		["Chimes_a", "A chime at the note of A."],
		["Chimes_ab", "A chime at the note of A Flat."],
		["Chimes_b", "A chime at the note of B."],
		["Chimes_bb", "A chime at the note of B Flat."],
		["Chimes_c", "A chime at the note of C."],
		["Chimes_cb", "A chime at the note of C Flat."],
		["Chimes_d", "A chime at the note of D."],
		["Chimes_e", "A chime at the note of E."],
		["Chimes_eb", "A chime at the note of E Flat."],
		["Chimes_f", "A chime at the note of F."],
		["Chimes_fb", "A chime at the note of F Flat."],
		["Chimes_g", "A chime at the note of G."]
	]
}
```

Appendix G : Sample Consent Form

‘MakeTheCosm’ Study Participation

Participant Consent Form

I can confirm that I voluntarily took part in the ‘MakeTheCosm’ study for Alexander Michael Roche’s Third Year Dissertation study, providing feedback on the usability of the project. I am aware that any element of my participation may be used within his dissertation, with the proviso that I remain anonymous. I am also aware of my right to withdraw myself from the study at any point, should I wish to.

Signed:_____ Date:_____

Appendix H : Work logs

Work logs can be found at http://github.com/AMRoche/MakeTheCosm for a full commit history of this project.

Appendix I : Full source code

Note : *MakeTheCosm refers to root directory of project.*

MakeTheCosm/index.html

```
<html>
	<header>
		<link rel="stylesheet" href="css/bootstrap.css" type="text/css">
	</header>
	<body>
		<div class="offset3 span8 well">
			<h2>Welcome to a third year project!</h2>
			<p>
				<h5>This is a short bit of text to masquerade as content. You should also probably know that this is still
				very heavily in development, and you should come back and check around May 2nd for the completed
				bit and piece!</h5>
			</p>

			<div>
				<a style="float:left;" class = "btn btn-large btn-primary"
```

```
href="makeyMakey">Link to the input page.</a>


                                <a style="float:right;"       class = "btn btn-large btn-primary"
href="mapping">Link to the map page.</a>
                                <a style="float:left;clear:both;" class = "btn btn-primary"
href="http://www.youtube.com/watch?v=HR8AHtdqaDQ">Link to "How To" video.</a>
                                <a style="float:right;" class = "btn btn-primary"
href="http://www.youtube.com/watch?v=qG-mHjIwtVs&feature=youtu.be">Link to "How To" video.</a>
                        </div>
                        <div style="clear:both;">
                                <div style="float:left;"class="label label-success">
                                        Mostly feature complete.
                                </div>
                                <div class="label label-success" style="float:right;">
                                        Mostly feature complete.Needs UX testing.
                                </div>
                        </div>
                </div>
        </body>
</html>
```

MakeTheCosm/css/bootstrap.css

http://twitter.github.io/bootstrap/

MakeTheCosm/makeyMakey.index.html

```
<!DOCTYPE html>
<html>
        <header>
                <script type="text/javascript" src="script/jquery164.min.js"></script>
                <script type="text/javascript" src="script/location.js"></script>
                <script type="text/javascript" src="script/formGenerator.js"></script>
                <script type="text/javascript" src="script/streaming.js"></script>
                <link rel="stylesheet" href="css/bootstrap.css" type="text/css">
                <script>
                        console.log("things loaded");
                </script>
        </header>
        <body>
                <div id="wrapper" class="span12 offset2">
```

```
                        <div class="span3">
                                <div id="streamingBanner"style="float:left;
margin-top:5px;display:none;"><span style="font-size:18px;">Streaming  : </span></div><div id="indicator"
style="margin-left:10px; float:left; width:30px; display:none; height:30px;border-radius:20px;
background-color:red;"></div>
                        </div>
                        <div class="span8">
                <!--    <button  class="btn"
onclick="(function(){document.getElementById('devConsole').style.display = '';})();">
                                Show DevConsole
                        </button>
                        <button  class="btn"
onclick="(function(){document.getElementById('devConsole').style.display = 'none';})();">
                                Hide DevConsole
                        </button>-->
                        </div>
                        <div class="span8"id="devConsole" style = "background:gray; display:none">
                                <div class="span8">
                                <!--interval = setInterval(testThing,750);-->
                                <button  class="btn" onclick="(function(){interval =
window.setInterval(testThing(),750)})();">
                                Start Test Transmission
                                </button>
                                <button  class="btn"
onclick="(function(){window.clearInterval(interval);console.log('Ceasing Testing.');})();">
                                Stop Test Transmission
                                </button>
                                <button  class="btn" onclick="(function(){newFeed();})();">
                                Add a New Feed
                                </button>
                </div>
                                <div class="span8">
                                <div class="form-horizontal">
                                <form id="form2" onsubmit = "return apiKeySet()">
                                API KEY :
                                <input type="text" id="apiDevAccess" value="">
                                <button class="btn" type="submit">Submit</button>
                                </form>
                                </div>
                                <div class="form-horizontal">
                                <form id="feedGetter" onsubmit = "return FeedGet()">
                                Feed Number :
                                <input type="text" id="feedNum" value="105431"></input>
                                <button class="btn" type="submit">Submit</button>

                                </form>
                                </div>
                                <div class="form-horizontal">
```

```
                    <form id="underscoreAdding" onsubmit = "return underScore()">
                        Name of Stuff :
                        <input type="text" id="FEEDid" value="things that move"></input>
                        <button class="btn" type="submit">Submit</button>

                    </form>
                    </div>
                    </div>
                </div>
                <div id="setup">
                </div>

                <div id="form" class="well span12">
                    <div class="form-horizontal">
                        <h3>Welcome to part of Make the Cosm!</h3>
                    <p>
                        You are about to be asked for your location information. All we
need this for is to plot your location.
                    </p>
                    <p>
                        Both a Cosm API key and Cosm account are required to use this
web application, as it retrieves information
                        from the Cosm web servers. To sign up for an account visit <a
href="http://www.cosm.com" target="_blank"> the Cosm homepage</a>.
                    </p>
                    <p>Once you have obtained an API key, fill in the data below and click on
'Submit'.</p>
                    <p><em>Warning : </em>Using one API key for two separate
MakeTheCosm instances will cause malfunctions.</p>
                    <form id="selector" onsubmit="return false;" style="margin-top:20px">
                        API KEY :
                        <input type="text" id="apiAccess">
                        Number of Inputs :
                        <input type="number" id="numOfFeeds" value=1 min=1 max=10>
                        <input class="btn" type="submit" onclick =
"(function(){if(apiValidation(document.getElementById('apiAccess').value)){initialSet();}else{document.getEle
mentById('APIfail').style.display='';}})();">
                    <span id="APIfail" class="label label-important"style="display:none;">Your
API Key has failed to validate or is missing. Please try another one.</span>
                    </form>
                    <h3>
                        Detailed user instructions
                    </h3>
                    <p> A short video introduction can be found <a
href="http://www.youtube.com/watch?v=HR8AHtdqaDQ" target="_blank">here</a>.</p>
                    <p>
                        To use a Makey Makey with this (or any other USB device
capable of simulating a keyboard), simply plug it in
```

```
                                        and wait for its' drivers to finish installing, then use as trigger.
                                    </p>
                                    <p>
                                        This application is based on an interactive form which lets the user
add or subtract inputs and then stream them
                                        to the Cosm data service for other people to access and use.
                                    </p>
                                    </div>
                                </div>
                            </div>
        </body>
        <footer>
        </footer>
</html>
```

MakeTheCosm/makeyMakey/css/bootstrap.css

http://twitter.github.io/bootstrap/

MakeTheCosm/makeyMakey/backend/creater.php

```
<?php
header('Content-type: application/json');
$apiKEY = $_GET['APIkey'];
$thing = '"X-ApiKey : '.$apiKEY.'"';
$theurl = 'http://api.cosm.com/v2/feeds';
$ch = curl_init($theurl);

curl_setopt($ch, CURLOPT_CUSTOMREQUEST, 'POST'); // -X
curl_setopt($ch, CURLOPT_BINARYTRANSFER, TRUE); // --data-binary
curl_setopt($ch, CURLOPT_HTTPHEADER, array("X-ApiKey:".$apiKEY)); // -H
curl_setopt($ch, CURLOPT_HTTP_VERSION, CURL_HTTP_VERSION_1_0); // -0
curl_setopt($ch, CURLOPT_POSTFIELDS, '{"title":"Newly Created Feed With PHP"}');
curl_setopt($ch, CURLOPT_RETURNTRANSFER, 1);
curl_setopt($ch, CURLOPT_VERBOSE, 1);
curl_setopt($ch, CURLOPT_HEADER, 1);
$response = curl_exec($ch);
$header_size = curl_getinfo($ch, CURLINFO_HEADER_SIZE);
$header = substr($response, 0, $header_size);
$headers = get_headers_from_curl_response($response);

if (isset($_GET['jsoncallback'])) {
        print $_GET['jsoncallback'] . '(' . JSON_encode($headers) . ')';
} else {
        print JSON_encode($headers);
}

function get_headers_from_curl_response($response)
```

```
{
    $headers = array();
    $header_text = substr($response, 0, strpos($response, "\r\n\r\n"));
    foreach (explode("\r\n", $header_text) as $i => $line)
        if ($i === 0)
            $headers['http_code'] = $line;
        else
        {
            list ($key, $value) = explode(': ', $line);
            $headers[$key] = $value;
        }
    return $headers;
}


?>
```

MakeTheCosm/makeyMakey/scripts/jquery164.min.js

http://ajax.googleapis.com/ajax/libs/jquery/1.6.4/jquery.min.js

MakeTheCosm/makeyMakey/scripts/fetch.php

```
<?php
file_put_contents("jquery164.min.js",file_get_contents("http://ajax.googleapis.com/ajax/libs/jquery/1.6.4/jquery.min.js"));
//I am so very, very, very lazy.
?>
```

MakeTheCosm/makeyMakey/scripts/formGenerator.js

```
//http://cosm.com/docs/v2/feed/create.html
var keys = [87, 65, 83, 68, 70, 71, 38, 40, 37, 39];
var chars = ['W', 'A', 'S', 'D', 'F', 'G', 'UP', 'DOWN', 'LEFT', 'RIGHT'];
var choices;
// ^ int holding how many options there are on screen.
var formSubmitted = [new Array(), new Array(), new Array(), "", ""];
// ^ inputId, inputKey, invert,feedId, feedTitle
var SendUrl;
var feedsToPush = {};
// <- Object for stuff to send to server.
var feedsValues = new Array();
//values in feed.
var toStream = false;
//controls whether to even log information.
var pushInterval = null;
//thing holding the setInterval for pushing information.
```

```
var delimiter = "¬!¬";
//delimiter for array holding stream ID's
var apiKEY = null;

function charGet(number) {
	for (var i = 0; i < keys.length; i++) {
		if (number == keys[i]) {
			return chars[i];
		}
	}
	return "unfound";
}

function scanSubmit(texticle) {
	for (var i = 0; i < formSubmitted[1].length; i++) {
		if (texticle == formSubmitted[1][i]) {
			if (formSubmitted[2][i] == true) {
				//write a one to feedsValues
				feedsValues[i] = 1;
			}
			if (formSubmitted[2][i] == false) {
				//write a zero to feedsValues
				feedsValues[i] = 0;
			}
			return true;
		}
	}
	return false;
}

function scanSubmitDown(texticle) {
	for (var i = 0; i < formSubmitted[1].length; i++) {
		if (texticle == formSubmitted[1][i]) {
			if (formSubmitted[2][i] == true) {
				//write a zero to feedsValues
				feedsValues[i] = 0;
			}
			if (formSubmitted[2][i] == false) {
				//write a one to feedsValues
				feedsValues[i] = 1;
			}
			return true;
		}
	}
	return false;
}

$(function() {
```

```
	$(document).keydown(function(e) {
		if (toStream == true) {

			scanSubmitDown(charGet(e.keyCode));
			console.log("KEYDOWN");
			console.log(feedsValues);
		}
	});
	$(document).keyup(function(e) {
		if (toStream == true) {
			scanSubmit(charGet(e.keyCode));
			console.log(feedsValues);
		}
	});

});

function apiKeySet() {
	apiKEY = document.getElementById("apiDevAccess").value;
	console.log("API key set to : " + apiKEY);
	console.log(apiValidation(apiKEY));
	return false;
}

function initialSet() {

	if(document.getElementById("apiAccess").value == "" ){
			apiKEY = prompt("Please enter a valid API key.","");
			if(apiKEY == ""){return false;}
			else if(apiKEY!=""&&apiKEY!=null){
			document.getElementById("apiAccess").value = apiKEY;
			initialSet();
			return false;}
	}
	if (document.getElementById("apiAccess").value != "") {
		console.log(document.getElementById("apiAccess").value);
		apiKEY = document.getElementById("apiAccess").value;
		console.log("api key changed to : " + apiKEY);
	document.getElementById("streamingBanner").style.display="inline-block";
	document.getElementById("indicator").style.display="inline-block";
	var optionLength = document.getElementById("numOfFeeds").value;
	choices = optionLength;
	var stringyThing = "";
	stringyThing += '<div id="extraButtons">';
	if (choices < 10) {
		stringyThing += '<input type="button" class="btn" id="addOpt" onclick="addOption(' +
choices + ')"value="[ + ]"\>';
	}
```

```
	if (choices > 1) {
		stringyThing += '<input type = "button" class="btn" id="removeOpt"
onclick="removeOption(' + choices + ')"value="[ - ]"\>';
	}
	stringyThing += '</div>';
	stringyThing += '<form id = "streamer" onsubmit = "return resultParser()">';
	stringyThing += 'Feed I.D. (leave blank to generate new feed) : <input type="text"
id="feedId"></input>';
	stringyThing += 'Feed Title : <input type="text" id="feedTitle"></input>';
	for (var i = 0; i < optionLength; i++) {
		stringyThing += '<div id="input' + i + '" class="inputs"><div class="sendIndicator"
style="float:left; width:30px; height:30px;border-radius:20px;margin-right:50px; background-color:black;
display:none;"></div>Name : <input type ="text" class="inputId"> Key to bind to : ' + optString(i) + ' Invert
Input : <input type="checkbox" data-toggle="buttons-checkbox" class="invert"
value="invert"></input></div>';
	}//inputId, inputKey invert,
	stringyThing += '<button type ="submit" class="btn"id="subInit">Start Streaming!</button><input
type="button" class="btn" id="streamStop" onclick = "(function(){stopStreaming();})();" value="Stop
Streaming"\>';
	stringyThing += '</form>';
	//document.getElementById("form").style.display = "none";
	document.getElementById("form").innerHTML = stringyThing;
	return false;
	}
}

function optString(input) {
	var optionString = "<select class='inputKey'>";
	optionString += "<option value='" + chars[input] + "'>" + chars[input] + "</option>";
	optionString += "<option value='---'>---</option>";
	for (var i = 0; i < 10; i++) {
		if (i != input) {
			optionString += "<option value='" + chars[i] + "'>" + chars[i] + "</option>";
		}
	}
	optionString += "</select>";
	return optionString;
}

function removeOption(currentCount) {
	//console.log(currentCount);
	choices--;
	if (choices < 2) {
		//console.log(document.getElementById("extraButtons"));
		var element = document.getElementById("removeOpt");
		element.parentNode.removeChild(element);
	}
	if (choices > 1) {
```

```
                document.getElementById("extraButtons").innerHTML = '<input type="button" class="btn"
id="addOpt" onclick="addOption(' + choices + ')" value="[ + ]"\>' + '<input type="button" class="btn"
id="removeOpt" onclick="removeOption(' + choices + ')" value="[ - ]"\>';
        }

        var element = document.getElementById("input" + choices);
        //console.log(element);
        element.parentNode.removeChild(element);

        document.getElementById("addOpt").onclick = function() {
                addOption(choices)
        };
        if (choices > 2) {
                document.getElementById("removeOpt").onclick = function() {
                        removeOption(choices)
                };
        }
}

function addOption(currentCount) {
        if (choices >= 9) {
                var element = document.getElementById("addOpt");
                element.parentNode.removeChild(element);
        } else {
                document.getElementById("extraButtons").innerHTML = '<input type="button" class="btn"
id="addOpt" onclick="addOption(' + choices + ')"value= "[ + ]"\>' + '<input type="button" class="btn"
id="removeOpt" onclick="removeOption(' + choices + ')"value="[ - ]">';
        }
        //console.log(choices);
        var remove = document.getElementById("streamStop");
        var element = document.getElementById("subInit");
        remove.parentNode.removeChild(remove);
        element.parentNode.removeChild(element);
        //inputId, inputKey invert,
        $('#streamer').append('<div id="input' + currentCount + '" class="inputs"><div class="sendIndicator"
style="float:left; width:30px; height:30px;border-radius:20px;margin-right:50px; background-color:black;
display:none;"></div>Name : <input type ="text" class="inputId"> Key to bind to : ' +
optString(currentCount) + ' Invert Input : <input type="checkbox" class="invert"
value="invert"></input></div>' + '<button type="submit" class="btn" id="subInit">Start
Streaming!</button><input type="button" class="btn" id="streamStop" onclick =
"(function(){stopStreaming();})();"value="Stop Streaming."\>');

        choices++;
        if (currentCount < 9) {
                document.getElementById("addOpt").onclick = function() {
                        addOption(choices)
                };
        }
```

```
	document.getElementById("removeOpt").onclick = function() {
		removeOption(choices)
	};
}
function apiValidation(supposedKey){
	var validKey = false;
	console.log("testing");
	$.ajax({
			type : "GET",
			url : "http://api.cosm.com/v2/" + "feeds",
			beforeSend : function(request) {
				request.setRequestHeader("X-ApiKey", supposedKey);
			},
			async : false,
			data : "{}",
			always : function() {
				console.log("done");
			},
			error : function(response) {
				console.log(response);
			},
			success : function(response) {validKey = true;},
			error : function(response){

					validKey = false;

				}
});
if(!validKey){return false;}
else{return true;}
}
function boolOrNot(thingy) {
	if (thingy == true) {
		return 1;
	} else {
		return 0;
	}
}

function resultParser() {
	toStream = true;
	var inputid, inputkey, Invert;
	formSubmitted[0].length = 0;
	formSubmitted[1].length = 0;
	formSubmitted[2].length = 0;
	formSubmitted[3] = "";
	feedsValues.length = 0;
	inputid = document.getElementsByClassName("inputId");
```

```
inputkey = document.getElementsByClassName("inputKey");
invert = document.getElementsByClassName("invert");
console.log(feedsValues + "." + inputid.length);

for (var i = 0; i < inputid.length; i++) {

		inputid[i].value = inputid[i].value.split(' ').join('_')
		if(!inputid[i].value.match(/^[0-9a-z]+$/)) {
			console.log("doing!");
	inputid[i].value = "Feed_"+i+"_input_id_not_alphanumeric";
   }

		formSubmitted[0].push(inputid[i].value);
		formSubmitted[1].push(inputkey[i].value);
		formSubmitted[2].push(invert[i].checked);
		feedsValues.push(boolOrNot(invert[i].checked));
}

/*pull id from stream suggested and input.
 * roll through stream id's and if case insensitive equal to input
 * make input the stream id found such that shit isn't fucked up.
 * use delimiter in found id's so we know it's not deleted.
 delete all stream id's which do not contain delimiter.
 */
if (document.getElementById("feedId").value == "") {
		toStream = false;
		addFeed();
}
else if (document.getElementById("feedId").value != "") {
		formSubmitted[3] = document.getElementById("feedId").value;
		formSubmitted[4] = document.getElementById("feedTitle").value;

		//all testing stuff goes here. Let's add stuff to the dev console for now.

		var fNum = document.getElementById("feedNum").value;
		var containArray = new Array();
		$.ajax({
			url : "http://api.cosm.com/v2/" + "feeds/" + fNum,
			beforeSend : function(request) {
				request.setRequestHeader("X-ApiKey", apiKEY);
			},
			data : "",
			always : function() {
				console.log("done");
			},
			error : function(response) {
				console.log(response);
			},
```

```
                        success : function(response) {
                                //      document.getElementById("spanner").innerHTML = response;

                                if (response.datastreams != undefined) {
                                        for (var i = 0; i < response.datastreams.length; i++) {
                                                containArray.push(response.datastreams[i].id);
                                        }

                                        for (var i = 0; i < containArray.length; i++) {
                                                for (var j = 0; j < formSubmitted[0].length; j++) {
                                                        if (formSubmitted[0][j].toLowerCase() ==
containArray[i].toLowerCase()) {
                                                                formSubmitted[0][i] = containArray[i];
                                                                containArray[i] += delimiter;
                                                        }
                                                }
                                        }
                                        for (var i = 0; i < containArray.length; i++) {
                                                if (containArray[i].indexOf(delimiter) == -1) {
                                                        console.log("deleting " + containArray[i]);
                                                        deleteElements(fNum, containArray[i]);
                                                }
                                        }
                                }
                                //      console.log(containArray);
                                //      console.log(checker);
                                pushInterval = setInterval(pushToServer, 660);
                        }
                });

                //end of testing here.

        }
        console.log(formSubmitted);
        return false;
}

function addFeed() {
        var urlLink = null;
                $.ajax({
                        type : "GET",
                        url : "backend/creater.php",
                        data : {
                                APIkey : apiKEY
                        },
                        async : false,
                        always : function() {
                                console.log("done");
```

```
                        },
                        dataType : "jsonp",
                        success : function(data, textStatus, xhr) {
                                console.log(textStatus);
                        },
                        error : function(response) {
                                console.log(response);
                        },
                        complete : function(response) {
                                console.log(response);
                                if (JSON.parse(response.responseText).Location == undefined) {
                                        addFeed();
                                } else {
                                        urlLink = JSON.parse(response.responseText).Location;
                                        urlLink = urlLink.split("/")[5];
                                        console.log(urlLink);
                                        formSubmitted[3] = urlLink;
                                        formSubmitted[4] = document.getElementById("feedTitle").value;
                                        toStream = true;
                                        document.getElementById("feedId").value = urlLink;
                                        if (formSubmitted[4].length == 0) {
                                                formSubmitted[4] = "A test title for you.";
                                        }
                                }

                        },
                        failure : function(response, status, xhr) {
                                console.log(response);
                                console.log(xhr.getAllResponseHeaders());
                        }
                });
        pushInterval = setInterval(pushToServer, 660);
}

function stopStreaming() {
        document.getElementById("indicator").style.backgroundColor = "red";
        for(var i = 0; i < document.getElementsByClassName("sendIndicator").length; i++){
                document.getElementsByClassName("sendIndicator")[i].style.display = "none";
        }
        toStream = false;
        console.log(toStream);
        if (pushInterval) {
                clearInterval(pushInterval);
        }
        console.log("cleared");
}

function pushToServer() {
```

```
	document.getElementById("indicator").style.backgroundColor = "green";
	//inputId, inputKey, invert,feedId, feedTitle
	//formSubmitted
	SendUrl = "http://api.cosm.com/v2/" + "feeds/" + formSubmitted[3] + "?_method=PUT";
	console.log(SendUrl+"----------");
	var info = new Array();
	for (var i = 0; i < formSubmitted[0].length; i++) {
		document.getElementsByClassName("sendIndicator")[i].style.display = "";
		if(feedsValues[i] != (formSubmitted[2][i])?1:0){
			document.getElementsByClassName("sendIndicator")[i].style.backgroundColor =
"rgb(51,105,255)";
		}
		if(feedsValues[i] == (formSubmitted[2][i])?1:0){
			document.getElementsByClassName("sendIndicator")[i].style.backgroundColor =
"black";
		}
		var ID = formSubmitted[0][i];
		info.push({
			"current_value" : JSON.stringify(feedsValues[i]),
			"id" : ID,
			"max_value" : "1.0",
			"min_value" : "0.0"
		});
	}

	feedsToPush["title"] = formSubmitted[4];
	feedsToPush["version"] = "1.0.0";
	feedsToPush["website"] = "http://www.amroche.co.uk/netChimes";
	feedsToPush["tags"] = new Array("Makey Makey", "NetChimes", "Net Chimes");
	feedsToPush["location"] = {};
	feedsToPush["location"]["lat"] = latitude;
	feedsToPush["location"]["lon"] = longitude;
	feedsToPush["location"]["domain"] = "physical";
	feedsToPush["location"]["name"] = "A Makey Makey being used.";
	feedsToPush["location"]["exposure"] = "indoor";
	feedsToPush["datastreams"] = info;

	$.ajax({
		type : "PUT",
		url : SendUrl,
		crossDomain : true,
		beforeSend : function(request) {
			request.overrideMimeType("text/plain; charset=utf-8");
			request.setRequestHeader("X-ApiKey", apiKEY);
		},
		data : JSON.stringify(feedsToPush),
		always : function() {
			console.log("done");
```

```
                },
                success : function(response) {
                        //      document.getElementById("spanner").innerHTML = response;
                        console.log(response);
                },
                error : function(response) {
                        console.log(response);
                }
        });
        console.log("cycled through things.");
        console.log(JSON.stringify(feedsToPush));
}
```

makeTheCosm/makeyMakey/scripts/location.js

```
var longitude = 0;
var latitude = 0;

navigator.geolocation.getCurrentPosition(GetLocation);
function GetLocation(location) {
  latitude = location.coords.latitude.toString().split(".");
  if(latitude[1].length >= 4){
  latitude = latitude[0]+"."+latitude[1].substring(0,4);
  }
  else{
  latitude = latitude[0]+"."+latitude[1];
  }
  longitude = location.coords.longitude.toString().split(".");
  if(longitude[1].length >= 4){
  longitude = longitude[0]+"."+longitude[1].substring(0,4);
  }
  else{
  longitude = longitude[0]+"."+longitude[1];
  }
  console.log("Got Co-ordinates to "+location.coords.accuracy+"accuracy.");
  console.log(latitude+","+longitude);
}
```

makeTheCosm/makeyMakey/scripts/streaming.js

```
// Set your API key first
//cosm.setKey( "LEXFCUfXTzzUiPhvL0tSAAkJFkWSAKxLQkVLSExCMHlZTT0g" );
// Get feed content

var interval;

// Get realtime updates on a datastream
```

```
//cosm.datastream.subscribe( "61916", "sine60", function( event, data ) {
// console.log( data.current_value ); // Logs value changes in realtime
//});

//cosm.feed.update("105431","20",function(){console.log("done");});
//"X-ApiKey: "+ apiKEY);
//testThing();
function newFeed() {
	var urlLink = null;
	if (apiKEY == null) {
		console.log("LOL NO API KEY");
		apiKEY = prompt("Please enter a valid API key.", "");
	}
	if (apiKEY != null) {
		$.ajax({
			type : "GET",
			url : "backend/creater.php",
			data : {
				APIkey : apiKEY
			},
			async : false,
			always : function() {
				console.log("done");
			},
			dataType : "jsonp",
			success : function(data, textStatus, xhr) {
				console.log(textStatus);
					if (JSON.parse(response.responseText).Location == undefined) {
					newFeed();
				} else {
					urlLink = JSON.parse(response.responseText).Location;
					urlLink = urlLink.split("/")[5];
					document.getElementById("feedId").value = urlLink;
				}

			},
			error : function(response) {
				console.log(response);
				newFeed();
			},
			complete : function(response) {
				if (JSON.parse(response.responseText).Location == undefined) {
					newFeed();
				} else {
					urlLink = JSON.parse(response.responseText).Location;
					urlLink = urlLink.split("/")[5];
					document.getElementById("feedId").value = urlLink;
				}
```

```
                        },
                        failure : function(response, status, xhr) {
                                console.log(response);
                                console.log(xhr.getAllResponseHeaders());
                        }
                });
        }
}

function testThing() {
        var apiKEY = "LEXFCUfXTzzUiPhvL0tSAAkJFkWSAKxLQkVLSExCMHlZTT0g";
        var thingOne = Math.floor(Math.random() * 14);
        var thingTwo = Math.floor(Math.random() * 25);
        var testJSON =
'{"title":"ITEST","version":"1.0.0","website":"http://www.amroche.co.uk/netChimes","tags":["Makey
Makey","NetChimes","Net Chimes"],"location":{"lat":"0","lon":"-59","domain":"physical","name":"A Makey
Makey being used.","exposure":"indoor"},' + '"datastreams":[' +
'{"id":"Thing_1","current_value":"1","max_value":"1.0","min_value":"0.0"},' +
'{"id":"Thing_2","current_value":"0","max_value":"1.0","min_value":"0.0"}' + ']} ';
        var sureJSON = '{"title":"netChimes web test, London.",' + '"version":"1.0.0",' + '"datastreams":[' +
'{"id":"wind1","current_value":' + thingOne + '},' + '{"id":"wind2","current_value":' + thingTwo + '}]}';

        console.log("pushing" + thingOne + ":" + thingTwo);
        $.ajax({
                type : "PUT",
                url : "http://api.cosm.com/v2/" + "feeds/" + "105431" + "?_method=PUT",
                beforeSend : function(request) {
                        request.setRequestHeader("X-ApiKey", apiKEY);
                },
                data : testJSON,
                always : function() {
                        console.log("done");
                },
                error : function(response) {
                        console.log(response);
                },
                success : function(response) {
                        //      document.getElementById("spanner").innerHTML = response;
                        console.log(response);
                        console.log(thingOne);
                }
        });
}

function deleteElements(fNum, Name) {

        $.ajax({
```

```
		type : "delete",
		url : "http://api.cosm.com/v2/" + "feeds/" + fNum + "/datastreams/" + Name,
		beforeSend : function(request) {
			request.setRequestHeader("X-ApiKey", apiKEY);
		},
		data : "",
		always : function() {
			console.log("done");
		},
		error : function(response) {
			console.log(response);
		},
		success : function(response) {
			console.log(response)
		}
	});

}

function underScore() {
	console.log(document.getElementById("FEEDid").value.split(' ').join('_'));
	return false;
}

function FeedGet() {
	var fNum = document.getElementById("feedNum").value;
	var containArray = new Array();
	$.ajax({
		url : "http://api.cosm.com/v2/" + "feeds/" + fNum,
		beforeSend : function(request) {
			request.setRequestHeader("X-ApiKey", apiKEY);
		},
		data : "",
		always : function() {
			console.log("done");
		},
		error : function(response) {
			console.log(response);
		},
		success : function(response) {
			//	document.getElementById("spanner").innerHTML = response;
			for (var i = 0; i < response.datastreams.length; i++) {
				containArray.push(response.datastreams[i].id);
			}
			var checker = new Array("thing_1", "thing2", "thing3");
			for (var i = 0; i < containArray.length; i++) {
				for (var j = 0; j < checker.length; j++) {
					if (checker[j].toLowerCase() == containArray[i].toLowerCase()) {
```

```
                                        checker[i] = containArray[i];
                                        containArray[i] += delimiter;
                                }
                        }
                }
                for (var i = 0; i < containArray.length; i++) {
                        if (containArray[i].indexOf(delimiter) == -1) {
                                console.log("deleting " + containArray[i]);
                                deleteElements(fNum, containArray[i]);
                        }
                }
                console.log(containArray);
                console.log(checker);

        }
    });
    return false;
}
```

MakeTheCosm/mapping/index.html

```
<!DOCTYPE html>
<html>
    <head>
        <title>Make the Cosm</title>
        <meta http-equiv="Content-Type" content="text/html; charset=utf-8">
        <link rel="stylesheet" href="styles.css" type="text/css">
        <link rel='stylesheet' href='http://api.tiles.mapbox.com/mapbox.js/v0.6.7/mapbox.css'>
        <script src="http://api.tiles.mapbox.com/mapbox.js/v0.6.7/mapbox.js"></script>
    </head>
    <body>
        <ul class="slideshow">
            <li class="slide active" id="slide-1">
                <div class="content">
                    <h1>Feed Selection</h1>
                    <!--cosm API -->
                    <div class="wrapper">
```

```
                                        <div id="mapWrapper">
                                        <div id="questionSetter" class="backEmphasis">
                                                <label>Search for : </label>
                                                <input type="text" id = "queryBox"
onchange="question = this.value"/>
                                        </div>
                                        <div id="renderSetter" class="backEmphasis">
                                                <label>Results shown : </label>
                                                <input type="number"
id="maxRenderSetter"onchange = "maxRender = this.value">
                                        </div>
                                        <div id="map"></div>
                                        </div>
                                        <div id="feedList"><h2 class="backEmphasis"> Click on
a marker for more information!</h2></div>
                                </div>
                        </div>
                </li>

                <li class="slide static-background" id="slide-2">
                        <div class="content">
                                <h1>Sound Mapping</h1>
                                <div class="wrapper">
                                        <div id="listOfFeeds">
                                                <h3>Overview of feeds</h3>
                                                <h5>Displaying all selected.</h5>
                                                <div id="quickInfoSummation">

                                                </div>
                                        </div>
                                        <div id="selectedFeeds">
                                                <ul id="selectedFeedsList">

                                                </ul>
                                        </div>
                                </div>
                        </div>
                        <div>

                        </div>
                </li>

                <li class="slide" id="slide-3">
                        <div class="content">
                                <h1>Sound Selection</h1>
                                <div class="wrapper">
                                        <div id="audioSelection">
                                                <div id="soundDescriptionPreview">
```

```
                                                <p>
                                                        No sound loaded yet.
                                                </p>
                                        </div>
                                        <div id="audioList">
                                        </div>
                                                <div id="previewButton">
                                                <input type="button" value="Sound not
Loaded"/>
                                        </div>
                                                <div id="pauseButton">
                                                <input type="button" value="Sound not
Loaded"/>
                                        </div>

                                                <div id="addToListButton">
                                                <input type="button" value="+"/>
                                        </div>
                                        <div id="removeFromListButton">
                                                <input type="button" value="-"/>
                                        </div>



                                </div>
                                <div id="audioZone">
                                        <ul id="audioZoneList">

                                        </ul>
                                </div>
                        <div id="audioWrapper">
                                <div id="audioTryoutWrapper">
                                        </div>
                                        <div id="audioListWrapper">
                                        </div>
                                </div>
                        </div>
                </div>
                </li>

        </ul>
    <div id="overlay">
        <div id="formVerifyContainer">

                        <h3>Welcome to part of Make the Cosm!</h3>
                        <p>
                                You are about to be asked for your location information. All we
need this for is to plot your location.
```

```
                                        </p>
                                        <p>
                                                Both a Cosm API key and Cosm account are required to use this
web application, as it retrieves information
                                                from the Cosm web servers. To sign up for an account visit <a
href="http://www.cosm.com" target="_blank"> the Cosm homepage</a>.
                                        </p>
                                        <p><em>Warning : </em>Using one API key for two separate
MakeTheCosm instances will cause malfunctions.</p>
                                <form id="formVerify" onsubmit="return false;">
                                        <div class="formRow">
                                                <label>API KEY : </label>
                                                <input type="text" id="apiAccess" value="">

<!--LEXFCUfXTzzUiPhvL0tSAAkJFkWSAKxLQkVLSExCMHlZTT0g-->
                                        </div>
                                        <div class="formRow">
                                                <label>Search Query : </label>
                                                <input type="text" id="apiQueryAccess" value="">
                                        </div>
                                        <div class="formRow">
                                                <button class="normal" type="submit" onclick =
"validateAPIkey()">Submit</button>
                                        </div>

                                                <span id="checkingAPIkey"></span>
                                                <span id="APIempty" class="label
label-important"style="display:none;">Your API Key has been left as empty.</span>
                                                <span id="APIfail" class="label
label-important"style="display:none;">Your API Key has failed to validate. Please try another one.</span>
                                                <span id="QUESTIONfail" class="label
label-important"style="display:none;">Your Query string is empty. Please try another one.</span>
                                        </form>
                                        <h3>
                                                Detailed user instructions
                                        </h3>
                                        <p> A short video introduction can be found <a
href="http://www.youtube.com/watch?v=qG-mHjIwtVs" target="_blank">here</a>.</p>
                                        <p> This application is based on three panels; a feed selection, sound
mapping and a sound selection panel.
                                                These can be navigated using the arrow keys or the buttons
marked with chevrons (&amp;laquo;) and (&amp;raquo;).</p>
                                        <p> The <em>feed selection</em> panel renders information to a map
which the user can then select from.
                                                Once selected, data is rendered in the <em>sound
mapping</em> panel.</p>
                                        <p> The <em>sound mapping</em> feed consists of selected data being
graphed and assigned to criteria.</p>
```

```
                                        <p> The <em>sound selection</em> is a user interface designed to allow
the user to easily select sounds.
                                                Once a new sound is selected, it is added to the list accessible in
the <em>sound mapping</em> panel.</p>
                </div>
        </div>
                <header>

                </header>

                <footer>

                </footer>

                <button type="button" id="prev-button" class="hidden">
                        &laquo;
                </button>
                <button type="button" id="next-button">
                        &raquo;
                </button>

        </body>
        <script type= "text/javascript" src="script/jquery164.min.js"></script>
        <script type="text/javascript" src = "script/transfers.js"></script>
        <script type="text/javascript" src="script/highcharts.js"></script>
        <script type="text/javascript" src="script/history.js"></script>
        <script type="text/javascript" src="script/maps.js"></script>
        <script type="text/javascript" src="script/audioSettings.js"></script>
        <script type="text/javascript" src="script/feedOrdering.js"></script>
        <script type="text/javascript" src="script/mapUpdater.js"></script>
        <script src="script/location.js"></script>

</html>
```

MakeTheCosm/mapping/css/styles.less

```
@slide-duration: 600ms;
@slide-easing: ease-out;

//
// Layouts
// Vars

@base: 32px;
@m-height: 16px;

.marker-popup{
  background:red;
```

```
  }
body{
      word-wrap:break-word;
  }
.hidden{
  display:none;
  visibility:hidden;
  width:0px;
  height:0px;
}
.big-text {
          font-size: 36px;
          line-height: 36px;
}
.medium-text {
          font-size: 18px;
          line-height: 24px;
}
.small-text {
          font-size: 14px;
          line-height: 16px;
}

.button-text {
          font-size: 14px;
          line-height: 16px;
}

//
#slide-1 {background: rgb(100,110,150);}
//#slide-1 { background: #7af5a5;background: url(images/london-sky.png) no-repeat center
center;background-size: cover;}
#slide-2 {background: rgb(150,160,200);}
//#slide-2 { background: green url('http://placekitten.com/g/800/800') center center no-repeat; }
#slide-3 { background:  rgb(200,210,250);}
#overlay{
  position: absolute;
  top:0px;
  left:0px;
  width:100%;
  height:100%;
  background-color:white;
  z-index:9999;
  background: rgb(100,110,150);
     #formVerifyContainer{

         width:400px;
         margin: 0 auto;
```

```
        overflow-y :auto;
        height:100%;
       p{
           text-align:justify;
          }
       a{
           color:white;
          }
        #formVerify{
           width:300px;
           margin : 5px auto;
           border-radius:10px;
           padding:10px 0;
           background-color:rgba(255,255,255,0.2);
           .formRow{
              input[type="text"]{
                 float:right;
                 }
                 label{
                    float:left;
                    margin-top:5px;
                    }
              clear:both;
              width:100%;

       }
       }
        .normal{
           display:block;
            margin:0 auto;
       position:relative;
           width:100px;
           height:20px;
           font-size:12px;
           line-height:12px;
           }
     }
  }
.slideshow {

          position: fixed;
          left: -100%;
          display: block;
          width: 300%;
          height: 100%;
          padding: 0;
          margin: 0;
```

```
.slide {
        position: absolute;
        top: 0; left: 0;
        z-index: 2;

        display: block;
        width: 33.333333%;
        height: 100%;

        -webkit-transform: translateX(0);
        -moz-transform: translateX(0);
        transform: translateX(0);

        -webkit-transition: -webkit-transform @slide-duration @slide-easing 0;
        -moz-transition: -moz-transform @slide-duration @slide-easing 0;
        transition: transform @slide-duration @slide-easing 0;

        .content {

        }

        &.active {

                -webkit-transform: translateX(100%);
                -moz-transform: translateX(100%);
                transform: translateX(100%);

        }
        &.active ~ .slide {

        -webkit-transform: translateX(200%);
                -moz-transform: translateX(200%);
                transform: translateX(200%);

        }

        &.static-background {

                z-index: 1;

                -webkit-transition-delay: @slide-duration;
                -moz-transition-delay: @slide-duration;
                transition-delay: @slide-duration;
                -webkit-transition-duration: 0;
                -moz-transition-duration: 0;
                transition-duration: 0;

                .content {
```

```
                -webkit-transform: translateX(-100%);
                -moz-transform: translateX(-100%);
                transform: translateX(-100%);
                -webkit-transition: -webkit-transform @slide-duration @slide-easing 0;
                -moz-transition: -moz-transform @slide-duration @slide-easing 0;
                transition: transform @slide-duration @slide-easing 0;

            }

            &.active {

                -webkit-transition-delay: 0;
                -moz-transition-delay: 0;
                transition-delay: 0;

                .content {

                    -webkit-transform: translateX(0);
                    -moz-transform: translateX(0);
                    transform: translateX(0);

                }

            }

        }

        &.active ~ .slide.static-background .content {

            -webkit-transform: translateX(100%);
            -moz-transform: translateX(100%);
            transform: translateX(100%);

        }

    }

}


html, body {
    padding: 0;
    margin: 0;
    font: normal 14px/16px "Helvetica Neue", Helvetica, Arial, sans-serif;
    color: #ffffff;
}
```

```
button {

}
button#prev-button { bottom: 20px; left: 20px;
   position: fixed;
        display: block;
        width: 50px;
        height: 50px;
        border-radius:50px;
   -o-border-radius:50px;
        font-size: 20px;
        font-weight: bold;
        text-align: center;
   z-index:9000;
}
button#next-button { right: 20px; bottom: 20px;
   position: fixed;
        display: block;
        width: 50px;
        height: 50px;
   border-radius:50px;
   -o-border-radius:50px;
                z-index:9000;
        font-size: 20px;
        font-weight: bold;
        text-align: center;
}

* {
  -moz-box-sizing: border-box;
  -webkit-box-sizing: border-box;
  box-sizing: border-box;
}

header {
        position: absolute;
        top: 60px;


        display: block;

        .header_wrapper {
                width: 470px;
        }

h1 {
                float: left;
                color: #ffffff;
```

```
                width: 390px;
                height: 65px;
                background-color: #f4692b;
                .big-text;
                margin: 0;
                padding: 32px 30px 68px 70px;
        }

        .tri_top {
                float: left;
                width: 0px;
                height: 0px;
                border-style: solid;
                border-width: 50px 50px 0 0;
                border-color: #f4692b transparent transparent transparent;
        }

        .tri_bottom {
                float: left;
                width: 0px;
                height: 0px;
                border-style: solid;
                border-width: 50px 0 0 50px;
                border-color: transparent transparent transparent #f4692b;
        }
}

.content {
   float:left;
   width:100%;
   height:100%;
   background:rgba(0,0,0,0.1);
        //width: 768px;
        padding: 10px;
        text-shadow: #999999 1px 1px 0px;
        * {
                margin: 0;
                padding: 0;
        //      width: 400px;
        }

        h1 {
                .big-text;
                margin-left: 25%;
      margin-bottom:10px;
                span {
                        clear: left;
                }
```

```
        }
        h2 {
                .medium-text;
      // margin-left: 25%;
        margin-bottom:10px;
                span {
                        clear: left;
                }
        }
        p {
                .small-text;
        }
.wrapper{
    height:90%;
    float:left;
    padding:20px;
    background:rgba(0,0,0,0.1);
    width:100%;
    border-radius:30px;
    min-width:810px;
    overflow-x:auto;
    .backEmphasis{
          background:rgba(255,255,255,0.1);
          border-radius:10px;
       }
#audioSelection{
    float:left;
    padding:1%;
    border-radius:15px;
    background:rgba(0,0,0,0.1);
    margin : 0 auto 20px 5%;
    width:90%;
     input[type="button"]{
          border-radius:15px;
          width:65px;
          height:65px;
          font-weight:bold;
         }
    #audioList{
       line-height:65px;
       float:left;
       }
    #removeFromListButton{
       margin-left: 15px;
       float:right;
       display:none;
       input[type="button"]{
       font-size: 50px;
```

```
    line-height: 50px;
    padding-bottom: 8px;
    background-color:rgb(255,255,255);
    border-color:rgb(235,235,235);

    }
     input[type="button"]:hover{
         background-color:rgb(250,250,250);
         border-color:rgb(215,215,215);

         }
    }
#pauseButton{
    margin-left: 15px;
    float:right;
    display:none;
input[type="button"]{   border-color:rgb(115,230,115);
background-color:rgb(135,255,135);
    }
input[type="button"]:hover{
      border-color:rgb(95,210,95);
      background-color:rgb(115,235,115);
}
    }
#previewButton{
    margin-left: 15px;
    display:none;
    float:right;

    input[type="button"]{
          font-size:50px;
        border-color:rgb(115,230,115);
background-color:rgb(135,255,135);
    }
input[type="button"]:hover{
      border-color:rgb(95,210,95);
      background-color:rgb(115,235,115);
}
}
#pauseButton{
    input[type="button"]{
     line-height: 20px;
    font-size: 50px;
    padding-bottom: 10px;
    border-color:rgb(230,115,115);
    background-color:rgb(255,135,135);
    }
    input[type="button"]:hover{
```

```
            border-color:rgb(210,95,95);
             background-color:rgb(235,115,115);
          }
        }
      #addToListButton{
        margin-left: 15px;
        float:right;
        display:none;
        input[type="button"]{
        font-size: 50px;
        background-color:rgb(255,255,255);
            border-color:rgb(235,235,235);
        }
          input[type="button"]:hover{
            background-color:rgb(250,250,250);
            border-color:rgb(215,215,215);
            }
        }
      #soundDescriptionPreview{

        float:left;
        height:100%;
        overflow-y:auto;
        width:300px;
        margin-right:10px;
        }
      }
  #audioWrapper{
  }
#audioZone{
      margin:0 auto;
      background:rgba(0,0,0,0.1);
      width:90%;
      height:76%;
      border-radius:15px;
      overflow-y:auto;
      #audioZoneList{
            width:100%;
            list-style:none;
            padding-left:1%;
              li{
              min-width:310px;
                border-radius:15px;
                float:left;
                width:31%;
                margin:15px 1%;
                padding:5px;
               input[type="button"]{
```

```
  width:20px;
  height:20px;
  border-radius:10px;
  margin-left:10px;
  font-weight:bold;
  }
background:rgba(0,0,0,0.1);
.playSound{
      float:right;
      border-color:rgb(115,230,115);
      background-color:rgb(135,255,135);
   }
   .playSound:hover{
      border-color:rgb(95,210,95);
        background-color:rgb(115,235,115);
   }
.pauseSound{

   margin-top:4px;
 line-height:8px;
   float:right;
    border-color:rgb(230,115,115);
   background-color:rgb(255,135,135);
   }
   .pauseSound:hover{
     border-color:rgb(210,95,95);
   background-color:rgb(235,115,115);
      }
   h3{
      float: left;

      }
.removeSoundFromList{
   margin-right:3px;
   width:15px;
   height:15px;
   background-color:rgb(220,220,220);
   border-width:2px;
   border-color:rgb(255,255,255);
   border-style:solid;
   border-radius:10px;
   float:right;
   font-size:15px;
   line-height:12px;
   padding-left:1px;
   }
   .removeSoundFromList:hover{
       background-color:rgb(200,200,200);
```

```
                    border-color:rgb(220,220,220);
                    cursor:default;
                   }
               .uniqueSoundDescription{
                  height:3em;
                  overflow-y:auto;
                  width:90%;
                  float:left;
                  clear:both;
                  }
            }
      }
   }
#selectedFeeds{
   float:left;
   background:rgba(0,0,0,0.1);
   width:70%;
         height:100%;
       border-radius:20px;
   overflow-y:auto;
      #selectedFeedsList{
         width:100%;

         list-style:none;
            li{
               float:left;
               width:100%;
               margin:5px 0 5px 0;
               background:rgba(0,0,0,0.1);
                border-radius:15px;
               .graphDataContainer{

                  width:100%;
                  float: left;
                     .graphInfo{
                        padding-left:5px;
                        width: 30%;
                        float: left;
                        h2{margin-top:5px;}
                        h3{margin-bottom:5px;}
                        .feedVitalStats{
                           margin-left:15%;
                           }
                        input[type="button"]{
                          border-radius:15px;
                           width:70px;
                           height:32px;
                           margin-top:5px;
```

```
                    float:right;
                    margin-right:10px;
                    }
                }
                .graphContainer{
                    float:left;
                    width:30%;
                    padding:10px 0 10px 0;
                    height:150px;
                }

.options{
                    margin-top:10px;
                    margin-bottom:10px;
                    float:left;
                    width:40%;
                    .graphSettings{
                        margin: 5px 0 5px 0;
                        width:100%;
                        float:left;
                        padding:0 5px 0 10px;
                        .triggerSelection{
                            float:right;
                        }
                        .thresholdValue{
                            float:right;
                        }
                        .soundOptions{
                            float:right;
                        }
                        .noiseCheckboxThing{
                            float:right;
                        }
                    }
                }
            }
        }
    }
 #listOfFeeds{
    border-radius:20px;
    h3{margin-left:15%;}
    h5{margin-left:15%}
    padding:1%;
    float:left;
    margin-right:5%;
     background:rgba(0,0,0,0.1);
     width:25%;
```

```
    height:100%;
     #quickInfoSummation{
        border-radius:10px;
        margin-top:10px;
        background :rgba(0,0,0,0.1);
     width:100%;
     height:89%;
     overflow-y:auto;
        ul{
           list-style:none;
           li{
                 .quickWrapper{
                    border-radius:10px;
                    padding:2%;
                     background-color:rgba(255,255,255,0.2);
                    float:left;
                    width:100%;
                    margin-top:5px;
                    margin-bottom:5px;
                    input[type="button"]{
                         border-radius:15px;
                           width:70px;

                       float:right;
                       border-radius:15px;
                       }
                       .infoWahn{
                          .dataId{}
                          .curVal{}
                          .maxVal{}
                          .minVal{}
                       }
                    }
              }
           }
      }
   }
#mapWrapper{
   border-radius:20px;
   float:left;
   background:rgba(0,0,0,0.1);
   width:70%;
   height:100%;
   padding:10px 1% ;
 #questionSetter{
  float:left;
  }
 #renderSetter{
```

```
  float:right;

  }
   #map{
      border-radius:20px;
 clear:both;
      height:95%;

      }
   }
#feedList{
   height:100%;
   padding:1%;
   float:left;
   margin-left:5%;
   border-radius:20px;
    background:rgba(0,0,0,0.1);
    width:25%;
    height:99%;
   h2{
      text-align:center;
      }
   #dataWrapper{
      height:100%;
      h3{margin-left:15%;height:1em; overflow:hidden;
      }
      h5{margin-left:15%;}
         #creator{
            margin-left:15%;
               a{
                  color:#ffffff;
               }
            }
         #feedDescription{
            overflow-y:auto;

            font-style:italic;
            height:20%;
            }
            #dataStreamDataList{
               margin-top:10px;
               height:67%;
               overflow-y:auto;
               ul{list-style:none;
                  li{
                     .chosen{
                        background-color:rgba(100,255,100,0.2);
                        }
```

```
                .listed{
                    background-color:rgba(255,255,255,0.5) !important;
                    }
                .dataWrapping{
                     background-color:rgba(255,255,255,0.2);
                     border-radius:10px;
                    float:left;
                    width:100%;
                    padding:2%;
                    margin-top:5px;
                    margin-bottom:5px;
                    .listItem{
                        width:60%;
                        float:left;
                        #id{font-weight:bold;}
                        #value{}
                        }

                    .ButtonSection{

                        float:right;
                            input[type="button"]{
                                border-radius:15px;
                                float:right;
                                width:70px;
                                height:32px;
                                #addToMusic{}
                                #removeFromMusic{}
                            }
                        }
                    }
                }
            }
        }
    }
}
```

MakeTheCosm/mapping/css/styles.css

```
/* CSS crunched with Crunch - http://crunchapp.net/ */
.marker-popup{background:red;}
body{word-wrap:break-word;}
.hidden{display:none;visibility:hidden;width:0px;height:0px;}
.big-text{font-size:36px;line-height:36px;}
.medium-text{font-size:18px;line-height:24px;}
```

```
.small-text{font-size:14px;line-height:16px;}
.button-text{font-size:14px;line-height:16px;}
#slide-1{background:#646e96;}
#slide-2{background:#96a0c8;}
#slide-3{background:#c8d2fa;}
#overlay{position:absolute;top:0px;left:0px;width:100%;height:100%;background-color:white;z-index:9999;ba
ckground:#646e96;}#overlay #formVerifyContainer{width:400px;margin:0
auto;overflow-y:auto;height:100%;}#overlay #formVerifyContainer p{text-align:justify;}
#overlay #formVerifyContainer a{color:white;}
#overlay #formVerifyContainer #formVerify{width:300px;margin:5px auto;border-radius:10px;padding:10px
0;background-color:rgba(255, 255, 255, 0.2);}#overlay #formVerifyContainer #formVerify
.formRow{clear:both;width:100%;}#overlay #formVerifyContainer #formVerify .formRow
input[type="text"]{float:right;}
#overlay #formVerifyContainer #formVerify .formRow label{float:left;margin-top:5px;}
#overlay #formVerifyContainer .normal{display:block;margin:0
auto;position:relative;width:100px;height:20px;font-size:12px;line-height:12px;}
.slideshow{position:fixed;left:-100%;display:block;width:300%;height:100%;padding:0;margin:0;}.slideshow
.slide{position:absolute;top:0;left:0;z-index:2;display:block;width:33.333333%;height:100%;-webkit-transform
:translateX(0);-moz-transform:translateX(0);transform:translateX(0);-webkit-transition:-webkit-transform
600ms ease-out 0;-moz-transition:-moz-transform 600ms ease-out 0;transition:transform 600ms ease-out 0;}
.slideshow
.slide.active{-webkit-transform:translateX(100%);-moz-transform:translateX(100%);transform:translateX(100%
);}
.slideshow
.slide.active~.slide{-webkit-transform:translateX(200%);-moz-transform:translateX(200%);transform:translateX
(200%);}
.slideshow
.slide.static-background{z-index:1;-webkit-transition-delay:600ms;-moz-transition-delay:600ms;transition-del
ay:600ms;-webkit-transition-duration:0;-moz-transition-duration:0;transition-duration:0;}.slideshow
.slide.static-background
.content{-webkit-transform:translateX(-100%);-moz-transform:translateX(-100%);transform:translateX(-100%);
-webkit-transition:-webkit-transform 600ms ease-out 0;-moz-transition:-moz-transform 600ms ease-out
0;transition:transform 600ms ease-out 0;}
.slideshow
.slide.static-background.active{-webkit-transition-delay:0;-moz-transition-delay:0;transition-delay:0;}.slidesho
w .slide.static-background.active
.content{-webkit-transform:translateX(0);-moz-transform:translateX(0);transform:translateX(0);}
.slideshow .slide.active~.slide.static-background
.content{-webkit-transform:translateX(100%);-moz-transform:translateX(100%);transform:translateX(100%);}
html,body{padding:0;margin:0;font:normal 14px/16px "Helvetica Neue",Helvetica,Arial,sans-serif;color:#ffffff;}
button#prev-button{bottom:20px;left:20px;position:fixed;display:block;width:50px;height:50px;border-radius:5
0px;-o-border-radius:50px;font-size:20px;font-weight:bold;text-align:center;z-index:9000;}
button#next-button{right:20px;bottom:20px;position:fixed;display:block;width:50px;height:50px;border-radius
:50px;-o-border-radius:50px;z-index:9000;font-size:20px;font-weight:bold;text-align:center;}
*{-moz-box-sizing:border-box;-webkit-box-sizing:border-box;box-sizing:border-box;}
header{position:absolute;top:60px;display:block;}header .header_wrapper{width:470px;}
header
h1{float:left;color:#ffffff;width:390px;height:65px;background-color:#f4692b;font-size:36px;line-height:36px;ma
```

```
rgin:0;padding:32px 30px 68px 70px;}
header .tri_top{float:left;width:0px;height:0px;border-style:solid;border-width:50px 50px 0
0;border-color:#f4692b transparent transparent transparent;}
header .tri_bottom{float:left;width:0px;height:0px;border-style:solid;border-width:50px 0 0
50px;border-color:transparent transparent transparent #f4692b;}
.content{float:left;width:100%;height:100%;background:rgba(0, 0, 0, 0.1);padding:10px;text-shadow:#999999
1px 1px 0px;}.content *{margin:0;padding:0;}
.content h1{font-size:36px;line-height:36px;margin-left:25%;margin-bottom:10px;}.content h1 span{clear:left;}
.content h2{font-size:18px;line-height:24px;margin-bottom:10px;}.content h2 span{clear:left;}
.content p{font-size:14px;line-height:16px;}
.content .wrapper{height:90%;float:left;padding:20px;background:rgba(0, 0, 0,
0.1);width:100%;border-radius:30px;min-width:810px;overflow-x:auto;}.content .wrapper
.backEmphasis{background:rgba(255, 255, 255, 0.1);border-radius:10px;}
.content .wrapper #audioSelection{float:left;padding:1%;border-radius:15px;background:rgba(0, 0, 0,
0.1);margin:0 auto 20px 5%;width:90%;}.content .wrapper #audioSelection
input[type="button"]{border-radius:15px;width:65px;height:65px;font-weight:bold;}
.content .wrapper #audioSelection #audioList{line-height:65px;float:left;}
.content .wrapper #audioSelection #removeFromListButton{margin-left:15px;float:right;display:none;}.content
.wrapper #audioSelection #removeFromListButton
input[type="button"]{font-size:50px;line-height:50px;padding-bottom:8px;background-color:#ffffff;border-color:
#ebebeb;}
.content .wrapper #audioSelection #removeFromListButton
input[type="button"]:hover{background-color:#fafafa;border-color:#d7d7d7;}
.content .wrapper #audioSelection #pauseButton{margin-left:15px;float:right;display:none;}.content .wrapper
#audioSelection #pauseButton input[type="button"]{border-color:#73e673;background-color:#87ff87;}
.content .wrapper #audioSelection #pauseButton
input[type="button"]:hover{border-color:#5fd25f;background-color:#73eb73;}
.content .wrapper #audioSelection #previewButton{margin-left:15px;display:none;float:right;}.content
.wrapper #audioSelection #previewButton
input[type="button"]{font-size:50px;border-color:#73e673;background-color:#87ff87;}
.content .wrapper #audioSelection #previewButton
input[type="button"]:hover{border-color:#5fd25f;background-color:#73eb73;}
.content .wrapper #audioSelection #pauseButton
input[type="button"]{line-height:20px;font-size:50px;padding-bottom:10px;border-color:#e67373;background-
color:#ff8787;}
.content .wrapper #audioSelection #pauseButton
input[type="button"]:hover{border-color:#d25f5f;background-color:#eb7373;}
.content .wrapper #audioSelection #addToListButton{margin-left:15px;float:right;display:none;}.content
.wrapper #audioSelection #addToListButton
input[type="button"]{font-size:50px;background-color:#ffffff;border-color:#ebebeb;}
.content .wrapper #audioSelection #addToListButton
input[type="button"]:hover{background-color:#fafafa;border-color:#d7d7d7;}
.content .wrapper #audioSelection
#soundDescriptionPreview{float:left;height:100%;overflow-y:auto;width:300px;margin-right:10px;}
.content .wrapper #audioZone{margin:0 auto;background:rgba(0, 0, 0,
0.1);width:90%;height:76%;border-radius:15px;overflow-y:auto;}.content .wrapper #audioZone
#audioZoneList{width:100%;list-style:none;padding-left:1%;}.content .wrapper #audioZone #audioZoneList
li{min-width:310px;border-radius:15px;float:left;width:31%;margin:15px 1%;padding:5px;background:rgba(0,
```

0, 0, 0.1);}.content .wrapper #audioZone #audioZoneList li
input[type="button"]{width:20px;height:20px;border-radius:10px;margin-left:10px;font-weight:bold;}
.content .wrapper #audioZone #audioZoneList li
.playSound{float:right;border-color:#73e673;background-color:#87ff87;}
.content .wrapper #audioZone #audioZoneList li
.playSound:hover{border-color:#5fd25f;background-color:#73eb73;}
.content .wrapper #audioZone #audioZoneList li
.pauseSound{margin-top:4px;line-height:8px;float:right;border-color:#e67373;background-color:#ff8787;}
.content .wrapper #audioZone #audioZoneList li
.pauseSound:hover{border-color:#d25f5f;background-color:#eb7373;}
.content .wrapper #audioZone #audioZoneList li h3{float:left;}
.content .wrapper #audioZone #audioZoneList li
.removeSoundFromList{margin-right:3px;width:15px;height:15px;background-color:#dcdcdc;border-width:2px
;border-color:#ffffff;border-style:solid;border-radius:10px;float:right;font-size:15px;line-height:12px;padding-left
:1px;}
.content .wrapper #audioZone #audioZoneList li
.removeSoundFromList:hover{background-color:#c8c8c8;border-color:#dcdcdc;cursor:default;}
.content .wrapper #audioZone #audioZoneList li
.uniqueSoundDescription{height:3em;overflow-y:auto;width:90%;float:left;clear:both;}
.content .wrapper #selectedFeeds{float:left;background:rgba(0, 0, 0,
0.1);width:70%;height:100%;border-radius:20px;overflow-y:auto;}.content .wrapper #selectedFeeds
#selectedFeedsList{width:100%;list-style:none;}.content .wrapper #selectedFeeds #selectedFeedsList
li{float:left;width:100%;margin:5px 0 5px 0;background:rgba(0, 0, 0, 0.1);border-radius:15px;}.content
.wrapper #selectedFeeds #selectedFeedsList li .graphDataContainer{width:100%;float:left;}.content .wrapper
#selectedFeeds #selectedFeedsList li .graphDataContainer
.graphInfo{padding-left:5px;width:30%;float:left;}.content .wrapper #selectedFeeds #selectedFeedsList li
.graphDataContainer .graphInfo h2{margin-top:5px;}
.content .wrapper #selectedFeeds #selectedFeedsList li .graphDataContainer .graphInfo
h3{margin-bottom:5px;}
.content .wrapper #selectedFeeds #selectedFeedsList li .graphDataContainer .graphInfo
.feedVitalStats{margin-left:15%;}
.content .wrapper #selectedFeeds #selectedFeedsList li .graphDataContainer .graphInfo
input[type="button"]{border-radius:15px;width:70px;height:32px;margin-top:5px;float:right;margin-right:10px;}
.content .wrapper #selectedFeeds #selectedFeedsList li .graphDataContainer
.graphContainer{float:left;width:30%;padding:10px 0 10px 0;height:150px;}
.content .wrapper #selectedFeeds #selectedFeedsList li .graphDataContainer
.options{margin-top:10px;margin-bottom:10px;float:left;width:40%;}.content .wrapper #selectedFeeds
#selectedFeedsList li .graphDataContainer .options .graphSettings{margin:5px 0 5px
0;width:100%;float:left;padding:0 5px 0 10px;}.content .wrapper #selectedFeeds #selectedFeedsList li
.graphDataContainer .options .graphSettings .triggerSelection{float:right;}
.content .wrapper #selectedFeeds #selectedFeedsList li .graphDataContainer .options .graphSettings
.thresholdValue{float:right;}
.content .wrapper #selectedFeeds #selectedFeedsList li .graphDataContainer .options .graphSettings
.soundOptions{float:right;}
.content .wrapper #selectedFeeds #selectedFeedsList li .graphDataContainer .options .graphSettings
.noiseCheckboxThing{float:right;}
.content .wrapper #listOfFeeds{border-radius:20px;padding:1%;float:left;margin-right:5%;background:rgba(0,
0, 0, 0.1);width:25%;height:100%;}.content .wrapper #listOfFeeds h3{margin-left:15%;}

```
.content .wrapper #listOfFeeds h5{margin-left:15%;}
.content .wrapper #listOfFeeds
#quickInfoSummation{border-radius:10px;margin-top:10px;background:rgba(0, 0, 0,
0.1);width:100%;height:89%;overflow-y:auto;}.content .wrapper #listOfFeeds #quickInfoSummation
ul{list-style:none;}.content .wrapper #listOfFeeds #quickInfoSummation ul li
.quickWrapper{border-radius:10px;padding:2%;background-color:rgba(255, 255, 255,
0.2);float:left;width:100%;margin-top:5px;margin-bottom:5px;}.content .wrapper #listOfFeeds
#quickInfoSummation ul li .quickWrapper input[type="button"]{width:70px;float:right;border-radius:15px;}
.content .wrapper #mapWrapper{border-radius:20px;float:left;background:rgba(0, 0, 0,
0.1);width:70%;height:100%;padding:10px 1% ;}.content .wrapper #mapWrapper #questionSetter{float:left;}
.content .wrapper #mapWrapper #renderSetter{float:right;}
.content .wrapper #mapWrapper #map{border-radius:20px;clear:both;height:95%;}
.content .wrapper
#feedList{height:100%;padding:1%;float:left;margin-left:5%;border-radius:20px;background:rgba(0, 0, 0,
0.1);width:25%;height:99%;}.content .wrapper #feedList h2{text-align:center;}
.content .wrapper #feedList #dataWrapper{height:100%;}.content .wrapper #feedList #dataWrapper
h3{margin-left:15%;height:1em;overflow:hidden;}
.content .wrapper #feedList #dataWrapper h5{margin-left:15%;}
.content .wrapper #feedList #dataWrapper #creator{margin-left:15%;}.content .wrapper #feedList
#dataWrapper #creator a{color:#ffffff;}
.content .wrapper #feedList #dataWrapper #feedDescription{overflow-y:auto;font-style:italic;height:20%;}
.content .wrapper #feedList #dataWrapper
#dataStreamDataList{margin-top:10px;height:67%;overflow-y:auto;}.content .wrapper #feedList
#dataWrapper #dataStreamDataList ul{list-style:none;}.content .wrapper #feedList #dataWrapper
#dataStreamDataList ul li .chosen{background-color:rgba(100, 255, 100, 0.2);}
.content .wrapper #feedList #dataWrapper #dataStreamDataList ul li .listed{background-color:rgba(255, 255,
255, 0.5) !important;}
.content .wrapper #feedList #dataWrapper #dataStreamDataList ul li
.dataWrapping{background-color:rgba(255, 255, 255,
0.2);border-radius:10px;float:left;width:100%;padding:2%;margin-top:5px;margin-bottom:5px;}.content
.wrapper #feedList #dataWrapper #dataStreamDataList ul li .dataWrapping
.listItem{width:60%;float:left;}.content .wrapper #feedList #dataWrapper #dataStreamDataList ul li
.dataWrapping .listItem #id{font-weight:bold;}
.content .wrapper #feedList #dataWrapper #dataStreamDataList ul li .dataWrapping
.ButtonSection{float:right;}.content .wrapper #feedList #dataWrapper #dataStreamDataList ul li
.dataWrapping .ButtonSection input[type="button"]{border-radius:15px;float:right;width:70px;height:32px;}
```

MakeTheCosm/mapping/audio

Folder is full of audio files visible in the Github Repo mentioned in Appendix A.

MakeTheCosm/mapping/backend/dataRetrieval.php

```
<?php
header('Content-type: application/json');
$apiKEY = $_GET['APIkey'];
$q = $_GET['question'];
$maxCount = $_GET['maxCount'];
//$maxCount = 50;
```

```
//$apiKEY = "JOxnIA8lNaXSQ1aTWFrG4lF6s9aSAKxEbERVNEE5NHZNQT0g";
//$q = "arduino";

$theurl = 'http://api.cosm.com/v2/feeds';
$returnJSON = array();
$ch = curl_init($theurl);
curl_setopt($ch, CURLOPT_CUSTOMREQUEST, 'GET');
// -X
curl_setopt($ch, CURLOPT_BINARYTRANSFER, TRUE);
// --data-binary
curl_setopt($ch, CURLOPT_HTTPHEADER, array("X-ApiKey:" . $apiKEY));
// -H
curl_setopt($ch, CURLOPT_HTTP_VERSION, CURL_HTTP_VERSION_1_0);
// -0
curl_setopt($ch, CURLOPT_POSTFIELDS, "per_page=100&q=" . $q . "&order=created_at&status=live");
curl_setopt($ch, CURLOPT_RETURNTRANSFER, 1);
curl_setopt($ch, CURLOPT_VERBOSE, 1);
curl_setopt($ch, CURLOPT_HEADER, 1);

$response = curl_exec($ch);
//echo $response;
$response_size = curl_getinfo($ch, CURLINFO_HEADER_SIZE);
$response = substr($response, $response_size);

$response = JSON_decode($response, true);
//print_r($response);
//print_r(sizeof($response["results"]));
if(array_key_exists("results",$response)){
if (sizeof($response["results"]) >= $maxCount) {
        while (sizeof($returnJSON) < $maxCount) {
                for ($i = 0; $i < sizeof($response["results"]); $i++) {
                        if (array_key_exists("location", $response["results"][$i])) {
                                if (array_key_exists("lon", $response["results"][$i]["location"]) &&
array_key_exists("lat", $response["results"][$i]["location"])) {
                                        if ($response["results"][$i]["location"]["lon"] != 0 &&
$response["results"][$i]["location"]["lat"] != 0) {
                                                if (sizeof($response["results"][$i]["datastreams"]) > 0) {
                                                        array_push($returnJSON,
$response["results"][$i]);
                                                        if (sizeof($returnJSON) >= $maxCount) {
                                                                break;
                                                        }
                                                }

                                        }
                                }
                        }
                }
```

```
        }
}

else{
                for ($i = 0; $i < sizeof($response["results"]); $i++) {
                        if (array_key_exists("location", $response["results"][$i])) {
                                if (array_key_exists("lon", $response["results"][$i]["location"]) &&
array_key_exists("lat", $response["results"][$i]["location"])) {
                                        if ($response["results"][$i]["location"]["lon"] != 0 &&
$response["results"][$i]["location"]["lat"] != 0) {
                                                if (sizeof($response["results"][$i]["datastreams"]) > 0) {
                                                        array_push($returnJSON,
$response["results"][$i]);
                                                }

                                        }
                                }
                        }
                }
}
}
//print_r($returnJSON);
if (isset($_GET['jsoncallback'])) {
        print $_GET['jsoncallback'] . '(' . JSON_encode($returnJSON) . ')';
} else {
        print JSON_encode($returnJSON);
}
?>
```

--MakeTheCosm/mapping/scripts/audioSettings.js

```
var chosenFeeds = {};
//information object for feeds that have been chosen.
var json;
//global information object
var BPM = 120;
var graphUpdateTime = 2;
//graph updates per second.
var graphResolution = 100;
//period in seconds to display on the graph.
var basePath = "audio/";
//base path to all of the files in question. folder structure is handled in the json file.
var soundFiles = loadJSON(basePath + "audioList.json");
var idPrefix = "addSound";
var songsSelected = [];

//add extra part in to global information object to be checked for to see if there is any listen requirements
```

```
already set.
//based on what is in the array at that point act on it in seperate method called every time there's a
successful ajax call
//returned to us. Should do it.
// <- link to HTML5 audio ->
http://www.position-absolute.com/articles/introduction-to-the-html5-audio-tag-javascript-manipulation/
//add a select box for the sound to trigger. That needs doing.
//save the sound selected to an array too in name form and, on cycling through it, have it played. Wham.

// onChange = (function(){setPlayArray(1,"+feedId+","+dataId+",this.value)});
//		chosenFeeds[feedId] array structure for each object.[dataId, currentVal, maxVal, minVal, GRAPH,
sonificationInformation[] ]);
// sonificationInformation[] structure :
//http://freesound.org/people/fins/sounds/?page=1#sound
setInterval(function() {
	checkForSongData();
}, 60000 / BPM);
setInterval(function() {
	graphUpdate();
}, 1000 / graphUpdateTime);
addAudioList();
//addAudioList(); //uncomment this once you've sorted it for the heirachy of your page. Shit works super fine
though.
//have audio dump to a hidden div for preview function, and then once added have it add to the div with the
list of stuff.
//list for the audio file. Include vital stats (maybe? Length? Description? Ask Brock?);

function addAudioList() {
	var fileTypes = soundFiles.opts;
	var stringToInsert = "<select id='addAudioMenu' onchange =
(function(){addAudio(document.getElementById('addAudioMenu').value)})();>"
	stringToInsert += "<option value = ">- Select an audio track -</option>";
	var indexing = 0;
	for (var key in soundFiles) {
		if (key == "init") {
			if (soundFiles[key].length > 0) {
				populateAudioList();
			}
		}
		if (key != "opts" && key != "init") {
			for (var i = 0; i < soundFiles[key].length; i++) {
				stringToInsert += "<option value = '" + i + "||" + key + "||" +
soundFiles[key][i][0] + "'>" + soundFiles[key][i][0].replace(/_/g, ' ') + "</option>";
			}
		}
	}
	stringToInsert += "</select>"
	document.getElementById("audioList").innerHTML = stringToInsert;
```

```
}

function addAudio(songToAdd) {
	var index = songToAdd.split("||")[0];
	var songFolder = songToAdd.split("||")[1];
	var songName = songToAdd.split("||")[2];
	if(songToAdd != ""){
		document.getElementById("removeFromListButton").style.display = "block";
		document.getElementById("addToListButton").style.display = "block";
		document.getElementById("pauseButton").style.display = "block";
		document.getElementById("previewButton").style.display = "block";
	}
	else{
		document.getElementById("removeFromListButton").style.display = "none";
		document.getElementById("addToListButton").style.display = "none";
		document.getElementById("pauseButton").style.display = "none";
		document.getElementById("previewButton").style.display = "none";
		document.getElementById("soundDescriptionPreview").innerHTML = "<p>Please select an
audio track.</p>";
	}
	if (songToAdd != "" && document.getElementById(idPrefix + songFolder + songName) ==
undefined) {
	//		songsSelected.push(songToAdd);
		//now to add audio element
		var insertString = "<audio class='hiddenAudioPlayers' id='" + idPrefix + songFolder +
songName + "'>";
		for (var i = 0; i < soundFiles.opts.length; i++) {
			////console.log(soundFiles.opts[i][0]);
			insertString += "<source src='" + basePath + songFolder + "/" + songName + "."
+ soundFiles.opts[i][0] + "' type='audio/" + soundFiles.opts[i][1] + "'>";
		}
		insertString += "</audio>";
		document.getElementById("audioTryoutWrapper").innerHTML = insertString;
	}
	if(document.getElementById(idPrefix+songFolder+songName)!=undefined){
		document.getElementById("previewButton").innerHTML = "<input value = '>'type = 'button'
onclick='document.getElementById(\"" + idPrefix + songFolder + songName + "\").play()'>";
		document.getElementById("pauseButton").innerHTML = "<input value = '||'type = 'button'
onclick='document.getElementById(\"" + idPrefix + songFolder + songName + "\").pause()'>";
		//console.log(soundFiles);
		//console.log(songFolder);
		document.getElementById("soundDescriptionPreview").innerHTML = "<p>" +
soundFiles[songFolder.toString()][index][1] + "</p>";
		//document.getElementById(idPrefix + songFolder + songName).play();
		document.getElementById("addToListButton").innerHTML = '<input type="button"
value="+"onclick="(function(){addAudioToList(document.getElementById(\'addAudioMenu\').value)})();"/>';
		document.getElementById("removeFromListButton").innerHTML = '<input type="button"
value="-"onclick="(function(){removeAudioFromList(document.getElementById(\'addAudioMenu\').value)})();"/
```

```
>';

        }
}
function updateAudioLists(type, info){
        var index = info.split("||")[0];
        var songFolder = info.split("||")[1];
        var songName = info.split("||")[2];
        var domList = document.getElementsByClassName("soundOptions");
        var vitalStats = document.getElementsByClassName("feedVitalStats");
        console.log(songName);
        if(type == "add"){
                for(var i = 0; i < domList.length; i++){
                        var element = document.createElement('option');
                        element.setAttribute('value',info);
                        element.appendChild(document.createTextNode(songName.replace(/_/g,' ')));
                        document.getElementsByClassName("soundOptions")[i].appendChild(element);
                }
        }
        if(type == "remove"){
                for(var i = 0; i < domList.length; i++){
                         var element = document.createElement('option');
                        element.setAttribute('value',songName);
                        element.appendChild(document.createTextNode(songName.replace(/_/g,' ')));
                        for(var j = 0; j < domList[i].length; j++){
                                if(document.getElementsByClassName("soundOptions")[i][j].value ==
info){

document.getElementsByClassName("soundOptions")[i].removeChild(document.getElementsByClassName(
"soundOptions")[i][j]);
                                }
                        }
                }
        }
}

function addAudioToList(songToAdd) {
        var index = songToAdd.split("||")[0];
        var songFolder = songToAdd.split("||")[1];
        var songName = songToAdd.split("||")[2];
        //console.log(document.getElementById("listItem" + songFolder + songName));
        if (document.getElementById("listItem" + songFolder + songName) == null) {
                songsSelected.push(songToAdd);
                var stringToAdd = "<li data-index='" + index + "' id='listItem" + songFolder + songName +
"'>";
                stringToAdd += '<h3 class="songName">' + songFolder + "://" + songName + " ~
"+songName.replace(/_/g, ' ')+"</h3>";
                        stringToAdd += '<div class="removeSoundFromList"
```

```
onclick="(function(){removeAudioFromList(\'' + songToAdd + '\');})();">×</div>';
                stringToAdd += '<div class = "uniqueSoundDescription" id="description' + songFolder +
songName + '"><p>' + soundFiles[songFolder.toString()][index][1] + '</p></div>';
                        stringToAdd += '<input type="button" class="pauseSound" value="||"
onclick="document.getElementById(\'' + idPrefix + songFolder + songName + '\').pause()" />';
        stringToAdd += '<input type="button" class="playSound" value=">"
onclick="document.getElementById(\'' + idPrefix + songFolder + songName + '\').play()" />';

                //ADD A BUTTON HERE WHICH LETS THE USER PLAY THE FRACKIN' SOUND.
                document.getElementById("audioZoneList").innerHTML += stringToAdd + "</li>";
                var insertString = "<audio class='hiddenAudioPlayers' id='" + idPrefix + songFolder +
songName + "'>";
                for (var i = 0; i < soundFiles.opts.length; i++) {
                        ////console.log(soundFiles.opts[i][0]);
                        insertString += "<source src='" + basePath + songFolder + "/" + songName + "."
+ soundFiles.opts[i][0] + "' type='audio/" + soundFiles.opts[i][1] + "'>";
                }
                insertString += "</audio>";
                document.getElementById("audioTryoutWrapper").innerHTML = insertString;
                document.getElementById("audioListWrapper").innerHTML +=
document.getElementById("audioTryoutWrapper").innerHTML;
                updateAudioLists("add",songToAdd);
        }
}

function removeAudioFromList(songToRemove) {
        //console.log(songToRemove);

        for(var i = 0; i < songsSelected.length; i++){
                if(songsSelected[i] == songToRemove){
                        songsSelected.splice(i,1);
                }
        }
        updateAudioLists("remove",songToRemove);
        var index = songToRemove.split("||")[0];
        var songFolder = songToRemove.split("||")[1].toString();
        var songName = songToRemove.split("||")[2].toString();
        if (document.getElementById("listItem" + songFolder + songName) != null) {
                var listEl = document.getElementById("listItem" + songFolder + songName);
                listEl.parentNode.removeChild(listEl);

                var element = document.getElementById(idPrefix + songFolder + songName);
                element.parentNode.removeChild(element);
        }
}

function graphUpdate() {
        for (var key in chosenFeeds) {
```

```
                for (var i = 0; i < chosenFeeds[key].length; i++) {
                        chosenFeeds[key][i][4].series[0].addPoint([(new Date()).getTime(),
parseFloat(chosenFeeds[key][i][1])], true, true);
                }
        }
}

function checkForSongData() {
        ////console.log(chosenFeeds);
        //DO ALL OF THE PLAYCHECKING HERE BITCHES!!!!
        for (var key in chosenFeeds) {
                for (var i = 0; i < chosenFeeds[key].length; i++) {
                        //"Greater>> Less<< Changes from Original=|="
                        var tester = chosenFeeds[key][i][5];
                        console.log(tester);
                        if (tester[0] != undefined && tester[1] != undefined && tester[2] != undefined &&
tester[3] == "true") {
                                //0 is mode, 1 is threshold, 2 is sound //checks values set in
audioSettings.js
                                //console.log("allDef'd");
                                //console.log(tester);
                                //console.log(tester[1]+"")
                                if (tester[0] == "=|=") {
                                        console.log("=|="+tester[1]+"||"+chosenFeeds[key][i][1]);
                                        if (tester[1] != chosenFeeds[key][i][1]) {

                                                playNoise(tester[2]);
                                                tester[1] = chosenFeeds[key][i][1];
                                        }
                                } else if (tester[0] == ">>") {
                                        console.log(">>"+tester[1]+"||"+chosenFeeds[key][i][1]);
                                        if (tester[1] > chosenFeeds[key][i][1]) {
                                                playNoise(tester[2]);
                                        }
                                } else if (tester[0] == "<<") {
                                        console.log("<<"+tester[1]+"||"+chosenFeeds[key][i][1]);
                                        if (tester[1] < chosenFeeds[key][i][1]) {
                                                playNoise(tester[2]);
                                        }
                                } else if (tester[0] == "==" || tester[0] == "|==|") {
                                        //console.log("|--|"+tester[1]+"||"+chosenFeeds[key][i][1]);
                                        if(tester[0] == "=="){
                                                playNoise(tester[2]);
                                        }
                                        if (tester[1] == chosenFeeds[key][i][1]) {
                                                playNoise(tester[2]);
                                        }
                                }
```

```
                        }
                }
        }
}

function getType(input) {
        var m = (/[\d]+(\.[\d]+)?/).exec(input);
        if (m) {
                // Check if there is a decimal place
                if (m[1]) {
                        return 'float';
                } else {
                        return 'int';
                }
        }
        return 'string';
}

function populateAudioList() {
        songsSelected = soundFiles["init"];
        for (var i = 0; i < songsSelected.length; i++) {
                //console.log("LOLZ");
                var toUse = songsSelected[i];
                var index = toUse.split("||")[0];
                var songFolder = toUse.split("||")[1];
                var songName = toUse.split("||")[2];
        var audioElement = document.createElement('audio');
        audioElement.setAttribute('class','hiddenAudioPlayers');
        audioElement.setAttribute('id', idPrefix + songFolder + songName);
        document.getElementById("audioListWrapper").appendChild(audioElement);
                var insertString = "";
                for (var j = 0; j < soundFiles.opts.length; j++) {
                        ////console.log(soundFiles.opts[i][0]);
                        insertString += "<source src='" + basePath + songFolder + "/" + songName + "."
+ soundFiles.opts[j][0] + "' type='audio/" + soundFiles.opts[j][1] + "'>";
                }
                document.getElementById(idPrefix + songFolder + songName).innerHTML = insertString;
                var listElement = document.createElement('li')
                listElement.setAttribute("data-index",index);
                listElement.setAttribute("id","listItem"+songFolder+songName);
                document.getElementById("audioZoneList").appendChild(listElement);
                insertString = "";
                insertString = '<h3 class="songName">' + songFolder + " :// " + songName + " ~
"+songName.replace(/_/g, ' ')+"</h3>";
                insertString += '<div class="removeSoundFromList"
onclick="(function(){removeAudioFromList(\'' + toUse + '\');})();">×</div>';
                insertString += '<div class = "uniqueSoundDescription" id="description' + songFolder +
songName + '"><p>' + soundFiles[songFolder.toString()][index][1] + '</p></div>';
```

```
                                insertString += '<input type="button" class="pauseSound" value="||"
onclick="document.getElementById(\'' + idPrefix + songFolder + songName + '\').pause()" />';
                        insertString += '<input type="button" class="playSound" value=">"
onclick="document.getElementById(\'' + idPrefix + songFolder + songName + '\').play()" />';

                        //ADD A BUTTON HERE WHICH LETS THE USER PLAY THE FRACKIN' SOUND.
                        document.getElementById("listItem"+songFolder+songName).innerHTML = insertString;

                }
                //songsSelected
                /*for(songsSelected.length){

                 }*/
}

function setPlayArray(index, feed, idOfFeed, value, currentVal) {
        //console.log(index + "," + feed);
        //console.log(idOfFeed);
        //console.log(value);
        index = parseInt(index);
        //if(index == 1){value = parseFloat(value);}
        //console.log(value);
        //chosenFeeds[feedId].push([dataId, currentVal, maxVal, minVal,new Array(6)]);
        for (var i = 0; i < chosenFeeds[feed].length; i++) {
                if (idOfFeed == chosenFeeds[feed][i][0]) {
                        //console.log("found");
                        if (index == 1) {
                                if (getType(value) == 'float' || getType(value) == 'int') {
                                        value = value.toString();
                                        if (value.indexOf(",") != -1) {
                                                value = value.replace(/\,/g, '.');
                                        }
                                        //float recognition fails on nums with string on end
                                        //int recognition fails with anything after int.
                                        chosenFeeds[feed][i][5][index] = parseFloat(value);
                                } else {
                                        //console.log("FAIL OF STRING VALUE BEOTCH");
                                }
                        } else {
                                value = value.toString().replace(/\s/g, "");
                                if(chosenFeeds[feed][i][5][index] == "=="){
                                        chosenFeeds[feed][i][5][1]=undefined;
                                }
                                if (value == "GreaterThan") {
                                        console.log("GREATER THAN EDITING");
                                        value = "<<";
                                } else if (value == "LessThan") {
                                        value = ">>";
```

```
                                } else if (value == "ValueChanges") {
                                        value = "=|=";
                                } else if (value == "IsEqualToCurrent") {
                                        value = "==";
                                } else if (value == "IsEqualTo(setThreshold)") {
                                        value = "|==|";
                                }
                                if (value == "=|=") {
                                        chosenFeeds[feed][i][5][1] = currentVal;
                                } else if (value == "==") {
                                        chosenFeeds[feed][i][5][1] = currentVal;
                                }
                                chosenFeeds[feed][i][5][index] = value;
                        }
                }
                console.log(chosenFeeds[feed][i][5]);
        }
}

function playNoise(songId) {
        var index = songId.split("||")[0];
        var songFolder = songId.split("||")[1];
        var songName = songId.split("||")[2];
        console.log(songId);
        document.getElementById(idPrefix + songFolder + songName).play();

}

function loadJSON(filePath) {
        // Load json file;
        var json = loadTextFileAjaxSync(filePath, "application/json");
        // Parse json
        return JSON.parse(json);
}

function loadTextFileAjaxSync(filePath, mimeType) {
        var xmlhttp = new XMLHttpRequest();
        xmlhttp.open("GET", filePath, false);
        if (mimeType != null) {
                if (xmlhttp.overrideMimeType) {
                        xmlhttp.overrideMimeType(mimeType);
                }
        }
        xmlhttp.send();
        if (xmlhttp.status == 200) {
                return xmlhttp.responseText;
        } else {
                // TODO Throw exception
```

```
                return null;
        }
}
```

MakeTheCosm/mapping/scripts/feedOrdering.js

```
function sonifyAdd(input) {
        var feedId = input.split(":")[0];
        var dataId = input.split(":")[1];
        var currentVal = input.split(":")[2];
        var maxVal = parseFloat(input.split(":")[3]);
        var minVal = parseFloat(input.split(":")[4]);
        console.log(maxVal);

document.getElementById(dataId).getElementsByClassName("ButtonSection")[0].childNodes[0].style.displa
y = "none";

document.getElementById(dataId).getElementsByClassName("ButtonSection")[0].childNodes[1].style.displa
y = "inline-block";
        document.getElementById(dataId).className = "dataWrapping listed";

        if (chosenFeeds[feedId] == undefined) {
                console.log("undefined");
                chosenFeeds[feedId] = new Array();
                chosenFeeds[feedId].push([dataId, currentVal, maxVal, minVal]);
        } else if (chosenFeeds[feedId] != undefined) {
                console.log("found" + dataId + "," + feedId);
                chosenFeeds[feedId].push([dataId, currentVal, maxVal, minVal]);
        }
        var index = chosenFeeds[feedId].length-1;
        console.log(chosenFeeds);
          var li = document.createElement('li');
        li.setAttribute('id',feedId+dataId+"sonNode");
        document.getElementById("selectedFeedsList").appendChild(li);
        var stringToInsert = "<div class = 'graphDataContainer' id='"+feedId+dataId+"container'>";
        stringToInsert += "<div class = 'graphInfo'>";
        stringToInsert += "<h2 class = 'feedVitalStats'>"+feedId+" :// "+dataId  + "</h2>";
        stringToInsert += "<h3> Value >> <span id='"+feedId+dataId+"liveValue'>" + currentVal +
"</span></h3>";
        stringToInsert += "<h3> Maxima >> <span id='"+feedId+dataId+"liveMax'>" + maxVal +
"</span></h3>";
        stringToInsert += "<h3> Minima >> <span id='"+feedId+dataId+"liveMin'>" + minVal +
"</span></h3>";
        stringToInsert += "<input type='button' value='Remove' style='display:inline-block;'
class='removeFromArrayGraphing' onclick='(function(){sonifyRemoveArray(\"" + feedId + ":" + dataId +
"\");})();'>";
        stringToInsert += "</div>";
```

```
		stringToInsert += "<div class='graphContainer' id='"+feedId + dataId +"graph'></div>";
		stringToInsert += "<div class='options' id='"+feedId + dataId +"options'>";
		stringToInsert += "<div class='graphSettings'><label class = 'graphVal'>Graph Value : </label>"
		stringToInsert += "<select class='triggerSelection'id='"+feedId + dataId +"select' onChange =
(function(){setPlayArray(0,'"+feedId+"','"+dataId+"',document.getElementById('"+feedId + dataId
+"select').value,"+chosenFeeds[feedId][index][1]+")})();>"
+"<option> -Select a trigger- </option>"
+"<option> Is Equal To Current </option>"
+"<option> Is Equal To (set Threshold) </option>"
+"<option> Greater Than </option>"
+"<option> Less Than </option>"
+"<option> Value Changes </option>"
+"</select></div>"
+"<div class='graphSettings'><label class = 'thresholdVal'>Threshold Value : </label><input
type='textarea'class='thresholdValue'id='"+feedId + dataId +"textArea'"
+"onChange = \"(function(){setPlayArray(1,'"+ feedId + "','" +dataId+"',document.getElementById('"+feedId +
dataId +"textArea').value,"+chosenFeeds[feedId][index][1]+")})();\"/>";

		stringToInsert += "</div><div class='graphSettings'><label class='trackVal'>Select a track :
</label><select class = 'soundOptions' id='"+feedId + dataId +"soundSelect' onChange =
(function(){setPlayArray(2,'"+feedId+"','"+dataId.toString()+"',document.getElementById('"+feedId + dataId
+"soundSelect').value,"+chosenFeeds[feedId][index][1]+")})();>"
+"<option> -Select a sound- </option>";
for(var i = 0; i < songsSelected.length; i++){
		stringToInsert += "<option
value='"+songsSelected[i]+"'>"+songsSelected[i].split("||")[2].replace(/_/g, ' ')+"</option>";
}

stringToInsert+="</select></div>";

		stringToInsert += "<graph class='graphSettings'><label class='noiseVal'>Make some noise :
</label><input type='checkbox' class='noiseCheckboxThing' id='"+feedId+dataId+"checkBox' onChange='"
+"(function(){setPlayArray(3,\""+feedId+"\",\""+dataId+"\",document.getElementById(\""+feedId + dataId
+"checkBox\").checked, null)})();'"
+" /> ";

		stringToInsert += "</div></div>";
		stringToInsert += "</div>";
		document.getElementById(feedId+dataId+"sonNode").innerHTML = stringToInsert;

		var chartMax = maxVal;
		var chartMin = minVal;

		if((chartMax-chartMin) <= 10){
			chartMax = chartMax + 25;
			chartMin = chartMin - 25;
		}
```

```
  chosenFeeds[feedId][chosenFeeds[feedId].length-1].push(
            new Highcharts.Chart({
        chart: {
           renderTo: document.getElementById(feedId + dataId +"graph"),
           type : 'area',
           marginRight: 10,
           events: {
              load: function() {
                 // set up the updating of the chart each second
                 var series = this.series[0];
              }
           }
        },
        title: false,

             credits: {
                enabled: false
             },
        xAxis: {
           type: 'datetime',
           tickPixelInterval: 150
        },
        yAxis: {
              max : chartMax,
              min : chartMin,
           title: false,
           allowDecimals : true,
           plotLines: [{
              value: 0,
              width: 1,
              color: '#808080'
           }]
        },
        tooltip: {
           formatter: function() {
               return Highcharts.dateFormat('%Y-%m-%d %H:%M:%S', this.x)
+'<br/>'+Highcharts.numberFormat(this.y, 2);
           }
        },
          plotOptions: {
           area: {
                          threshold : -999999,
              marker: {
                 enabled: false,
                 symbol: 'circle',
                 radius: 2,
                 states: {
```

```
                    hover: {
                        enabled: true
                    }
                }
            }
        }
    },
    legend: {
        enabled: false
    },
    exporting: {
        enabled: false
    },
    series: [{
        name: 'Streamed Data',
        data: (function() {
            // generate an array of random data
            var data = [],
                time = (new Date()).getTime(),
                i;
            for (i = graphUpdateTime * graphResolution * -1; i <= 0; i++) {
                data.push({
                    x: time + i * 1000,
                    y: -999999
                });
            }
            return data;
        })()
    }]
  })
, new Array(6)
//  chosenFeeds[feedId][chosenFeeds[feedId].length-1]);
      );



      displayListUpdate();

//document.getElementById(dataId).getElementsByClassName("ButtonSection")[0].childNodes.item("remov
eFromMusic").style.display = "inline-block";
      //start here tomorrow.
}

function sonifyRemove(input) {
      //.splice(2,1)

      var feedId = input.split(":")[0];
      var dataId = input.split(":")[1];
```

```
document.getElementById(dataId).getElementsByClassName("ButtonSection")[0].childNodes[0].style.displa
y = "inline-block";

document.getElementById(dataId).getElementsByClassName("ButtonSection")[0].childNodes[1].style.displa
y = "none";
	console.log(input);
	element = document.getElementById(feedId + dataId + "sonNode");//these two lines remove the
graph node element.
	element.parentNode.removeChild(element);
	document.getElementById(dataId).className = "dataWrapping";
	for (var i = 0; i < chosenFeeds[feedId].length; i++) {
		if (chosenFeeds[feedId][i][0] == dataId) {
			chosenFeeds[feedId].splice(i, 1);
			console.log(chosenFeeds[feedId]);
		}
	}
	if (chosenFeeds[feedId].length == 0) {
		delete chosenFeeds[feedId];
	}

	displayListUpdate();
}

function sonifyRemoveArray(input) {
var FeedId = input.split(":")[0];
var StreamId = input.split(":")[1];
	if(document.getElementById("uniqueDataFeedDisplayIdentifier").innerHTML == FeedId){
		document.getElementById(StreamId).className = "dataWrapping";

document.getElementById(StreamId).getElementsByClassName("ButtonSection")[0].childNodes[0].style.dis
play = "inline-block";

document.getElementById(StreamId).getElementsByClassName("ButtonSection")[0].childNodes[1].style.dis
play = "none";
	}
element = document.getElementById(FeedId + StreamId + "sonNode");//these two lines remove the graph
node element.
element.parentNode.removeChild(element);

	for (var i = 0; i < chosenFeeds[FeedId].length; i++) {
		if (chosenFeeds[FeedId][i][0] == StreamId) {
			chosenFeeds[FeedId].splice(i, 1);
			console.log(chosenFeeds[FeedId]);
		}
	}
	if (chosenFeeds[FeedId].length == 0) {
		delete chosenFeeds[FeedId];
```

```
		}
		displayListUpdate();
}

function displayListUpdate() {
		console.log("running!");
		var insertThing = "<ul>";

		for (var key in chosenFeeds) {
			for (var i = 0; i < chosenFeeds[key].length; i++) {
				insertThing += "<li>";
				insertThing += "<div class='quickWrapper'>";
				insertThing += "<div class='infoWahn'>";
				insertThing += "<div class='dataId'><h3>"+key + " :// " + chosenFeeds[key][i][0] +
"</h3></div>";
				insertThing += "<div class='curVal'>Value >> " + chosenFeeds[key][i][1] +
"</div>";
				insertThing += "<div class='maxVal'>Maxima >> " + chosenFeeds[key][i][2] +
"</div>";
				insertThing += "<div class='minVal'>Minima >> " + chosenFeeds[key][i][3] +
"</div>";
				insertThing += "</div>";
				insertThing += "<input type='button' value='Remove' style='display:inline-block;'
class='removeFromArray' onclick='(function(){sonifyRemoveArray(\"" + key + ":" + chosenFeeds[key][i][0] +
"\");})();'>";
				insertThing += "</div>";
			}
		}

		insertThing += "</ul>";
		document.getElementById("quickInfoSummation").innerHTML = insertThing;
}
```

MakeTheCosm/mapping/scripts/fetch.php

```
<?php
file_put_contents("jquery164.min.js",file_get_contents("http://ajax.googleapis.com/ajax/libs/jquery/1.6.4/jque
ry.min.js"));
//I am so very, very, very lazy.
?>
```

MakeTheCosm/mapping/scripts/highcharts.js

Obtainable from www.highcharts.com

MakeTheCosm/mapping/scripts/history.js

```
if(supports_history_api()){
		console.log("history!");
```

```
}
else{
        console.log("No history support :( ");
}
function supports_history_api() {
  return !!(window.history && history.pushState);
}
```

MakeTheCosm/mapping/scripts/jquery164.min.js

Obtainable at http://ajax.googleapis.com/ajax/libs/jquery/1.6.4/jquery.min.js

MakeTheCosm/mapping/scripts/location.js

```
var longitude = 0;
var latitude = 0;

navigator.geolocation.getCurrentPosition(GetLocation);
function GetLocation(location) {
        console.log(location);
   latitude = location.coords.latitude.toString().split(".");
   if(latitude[1].length >= 4){
   latitude = latitude[0]+"."+latitude[1].substring(0,4);
   }
   else{
   latitude = latitude[0]+"."+latitude[1];
   }
   longitude = location.coords.longitude.toString().split(".");
   if(longitude[1].length >= 4){
   longitude = longitude[0]+"."+longitude[1].substring(0,4);
   }
   else{
   longitude = longitude[0]+"."+longitude[1];
   }
   console.log("Got Co-ordinates to "+location.coords.accuracy+"accuracy.");
  // return [latitude+","+longitude];
  userLayer.add_feature({
                                        'geometry' : {
                                                'coordinates' : [longitude, latitude]
                                        },
                                        'properties' : {
                                                // https://github.com/mapbox/simplestyle-spec
                                                'marker-color' : userColour,
                                                'marker-symbol' : 'town-hall'
                                        }
                                });
}
```

MakeTheCosm/mapping/scripts/maps.js

```
var colour = "#32CDB4"; //flags
var userColour = "#550055";//home
var layer = mapbox.layer().id('examples.map-vyofok3q');
var userLayer;

var map = mapbox.map('map',layer,null);

map.zoom(2).center({
            lat : 20,
            lon : 0
});

userLayer = mapbox.markers.layer();
mapbox.markers.interaction(userLayer);
map.addLayer(userLayer);
layer = mapbox.markers.layer();
var interacting = mapbox.markers.interaction(layer);
map.addLayer(layer);
map.ui.zoomer.add();

layer.factory(function(feature) {
    var elem = mapbox.markers.simplestyle_factory(feature);
    MM.addEvent(elem, 'click', function(e) {
      var o = '<h3>'+feature.properties.feedName + '</h3>' +
        '<ul>';
        for(var i = 0; i < feature.properties.datastreams.length; i++){
                    o +=
"<li>"+feature.properties.datastreams[i].id+":"+feature.properties.datastreams[i].current_value;
        }
                    o+="</ul>";
                    e.innerHTML = o;
      e.stopPropagation();
    });
   return elem;
});

 interacting.formatter(function(feature) {
      var o = '<h3><span id="feedId" style="display:none;">'+feature.properties.id+"</span><span
id='feedName'>"+feature.properties.feedName + '</span></h3>' +
          '<ul>';
                              var thing = 3;
           if(feature.properties.datastreams.length < 3){var thing = feature.properties.datastreams.length;}
           if(feature.properties.datastreams.length > 3){var thing = 2;}
          for(var i = 0; i < thing; i++){
                    o +=
"<li>"+feature.properties.datastreams[i].id+":"+feature.properties.datastreams[i].current_value;
          }
```

```
        if(feature.properties.datastreams.length>3){
                o+= "<li>-> More than 3 inputs <-</li>";
        }
                        o+="</ul>";
    return o;
  });
var alert = document.getElementById('feedList');
console.log(alert);
```

MakeTheCosm/mapping/scripts/mapUpdater.js

```
//var apiKEY = "JOxnIA8lNaXSQ1aTWFrG4lF6s9aSAKxEbERVNEE5NHZNQT0g";
var apiKEY;
var question = "arduino";
var maxSize = 30;
var maxRender = 5;

//startStuff();
//validateAPIkey();
function startStuff() {
        document.getElementById("queryBox").value = question;
        document.getElementById("maxRenderSetter").value = maxRender;
        console.log("initiated");
        getInfo(question);
        setInterval(function() {
                getInfo(question)
        }, 660);
        getInfo(question);
        //if you need strictly data things updating, update them in getInfo. Else, go find checkForSongData
and graph updating.
        //getInfo(question);
}

function validateAPIkey() {

        if(document.getElementById('apiAccess').value.length == 0){
                //console.log("APILENGTH");
                document.getElementById('APIempty').style.display = 'inline-block';
                document.getElementById('QUESTIONfail').style.display = 'none';
                                        document.getElementById('APIfail').style.display = 'none';

        }
        else if(document.getElementById('apiQueryAccess').value.length == 0){
                //console.log("QUESTION");
                document.getElementById('QUESTIONfail').style.display = 'inline-block';
                document.getElementById('APIfail').style.display = 'none';
                                        document.getElementById('APIempty').style.display = 'none';
```

```
	}
	else{
		//console.log("stuff");
		if (apiValidation(document.getElementById('apiAccess').value)) {
			apiKEY = document.getElementById('apiAccess').value;
			question = document.getElementById('apiQueryAccess').value;
			document.getElementById('APIfail').style.display = 'none';
			document.getElementById('overlay').style.display = 'none';
			document.getElementById('APIempty').style.display = 'none';
			startStuff();
		}
		else{
			document.getElementById('APIfail').style.display = 'inline-block';
			document.getElementById('QUESTIONfail').style.display = 'none';
			document.getElementById('APIempty').style.display = 'none';

		}
	}
}
function apiValidation(supposedKey) {
	var validKey = false;
	//console.log("testing");
	$.ajax({
		type : "GET",
		url : "http://api.cosm.com/v2/" + "feeds",
		beforeSend : function(request) {
			request.setRequestHeader("X-ApiKey", supposedKey);
		},
		async : false,
		data : "{}",
		always : function() {
			//console.log("done");
		},
		error : function(response) {
			//console.log(response);
		},
		success : function(response) {
			validKey = true;
		},
		error : function(response) {

			validKey = false;

		}
	});
	if (!validKey) {
		//console.log("false");
		return false;
```

```
	} else {
		//console.log("true");
		return true;
	}
}

function getInfo(query) {
	$.ajax({
		type : "GET",
		url : "backend/dataRetrieval.php",
		crossDomain : true,
		beforeSend : function(request) {
			//request.overrideMimeType("text/plain; charset=utf-8");
			//request.setRequestHeader("X-ApiKey", apiKEY);
		},
		//$apiKEY = $_GET['APIkey'];
		//$q = $_GET['question'];
		//$maxCount = $_GET['maxCount'];
		async : true,
		data : {
			"APIkey" : apiKEY,
			"question" : query,
			"maxCount" : maxRender.toString()
		},
		always : function() {
			onsole.log("done");
		},
		success : function(response) {
			console.log(response);
			json = new Array();
			layer.features([]);
			for (var i = 0; i < response.length; i++) {
				//////console.log(response.results[i]);
				json.push(response[i]);
				layer.add_feature({
					'geometry' : {
						'coordinates' : [response[i].location.lon,
response[i].location.lat]
					},
					'properties' : {
						'marker-color' : colour,
						'marker-symbol' : 'embassy',
						'feedName' : response[i].title,
						'desc' : response[i].description,
						'datastreams' : response[i].datastreams,
						'id' : response[i].id
					}
				});
```

```
                        }
                        var checkId = document.getElementById("feedList").getAttribute("feedId");
                        if (checkId != undefined && checkId != "") {
                                updateInfo(checkId);
                        }
                        //                              layer.features(layer.markers());
                        var markers = document.getElementsByClassName('simplestyle-marker');
                        for (var i = 0; i < markers.length; i++) {
                                markers[i].addEventListener("click", getPertinentInfo);
                        }
                },
                error : function(response) {
                console.log(response);
                }
        });

}

function updateInfo(idOfDiv) {
        ////console.log("being called");
        if (document.getElementById("feedId") != null) {
                var idOfThing = document.getElementById("feedId").innerHTML;
        }
        for (var i = 0; i < json.length; i++) {

                if (chosenFeeds[json[i].id] != undefined) {
                        for (var dataI = 0; dataI < json[i].datastreams.length; dataI++) {
                                for (var IJSON = 0; IJSON < chosenFeeds[json[i].id].length; IJSON++) {
                                        if (json[i].datastreams[dataI].id ==
chosenFeeds[json[i].id][IJSON][0]) {
                                                chosenFeeds[json[i].id][IJSON][1] =
json[i].datastreams[dataI].current_value;
                                                //value   //json[i].datastreams[j].current_value
                                                chosenFeeds[json[i].id][IJSON][2] =
json[i].datastreams[dataI].max_value;
                                                //maxima
                                                chosenFeeds[json[i].id][IJSON][3] =
json[i].datastreams[dataI].min_value;
                                                //minima
                                                //console.log(json[i].id +
chosenFeeds[json[i].id][IJSON][0] + "liveValue");
                                                document.getElementById(json[i].id +
chosenFeeds[json[i].id][IJSON][0] + "liveValue").innerHTML = chosenFeeds[json[i].id][IJSON][1]
                                                document.getElementById(json[i].id +
chosenFeeds[json[i].id][IJSON][0] + "liveMax").innerHTML = chosenFeeds[json[i].id][IJSON][2]
                                                document.getElementById(json[i].id +
chosenFeeds[json[i].id][IJSON][0] + "liveMin").innerHTML = chosenFeeds[json[i].id][IJSON][3]
                                                //liveValue. liveMin, liveMax
```

```
                                                ////console.log(chosenFeeds[json[i].id][IJSON][6]);
                                                //////console.log("updating
"+chosenFeeds[json[i].id][IJSON][0]);
                                                //          chosenFeeds[json[i].id][IJSON][4].series[0]
                                                ////console.log(chosenFeeds[json[i].id][IJSON][5]);


//chosenFeeds[json[i].id][IJSON][4].series[0].data.push([(new
Date()).getTime(),json[i].datastreams[dataI].current_value]);
                                        }
                                }
                        }
                }

                if (idOfThing == json[i].id) {
                        ////console.log(json[i]);
                        var string = "<div id='dataWrapper'>";
                        string += "<h3>" + json[i].title + "</h3><h5 id='uniqueDataFeedDisplayIdentifier'>"
+ json[i].id + "</h5>";
                        string += "<p id='creator'>Creator : <a href='" + json[i].creator + "'target='_blank'>"
+ json[i].creator.substring(23, json[i].creator.length) + "</a></p>";
                        ////console.log();
                        if (json[i].description != undefined) {
                                string += "<p id='feedDescription'>" + json[i].description + "</p>";
                        }
                        else if(json[i].description == undefined){
                                string += "<p id='feedDescription'></p>";
                        }
                        string += "<div id='dataStreamDataList'>";
                        string += "<ul>";

                        for (var j = 0; j < json[i].datastreams.length; j++) {
                                var listed = false;
                                //chosenFeeds[feedId].push([dataId, currentVal]) ;
                                if (chosenFeeds[json[i].id] == undefined) {
                                        string += "<li><div class='dataWrapping' id='" +
json[i].datastreams[j].id + "'><div class='listItem' >";
                                } else if (chosenFeeds[json[i].id] != undefined) {
                                        var found = false;
                                        //for loop to see if found.
                                        for (var zed = 0; zed < chosenFeeds[json[i].id].length; zed++) {
                                                if (chosenFeeds[json[i].id][zed][0] ==
json[i].datastreams[j].id) {
                                                        found = true;
                                                }
                                        }

                                        if (found == false) {
```

```
                                                string += "<li><div class='dataWrapping' id='" +
json[i].datastreams[j].id + "'><div class='listItem' >";
                                        }
                                        if (found != false) {
                                                listed = true;
                                                string += "<li><div class='dataWrapping listed' id='" +
json[i].datastreams[j].id + "'><div class='listItem' >";
                                        }
                                }
                                string += "<p id='id'>" + json[i].datastreams[j].id + "</p>";
                                string += "<p>VALUE : <span id='value' data-val='" +
json[i].datastreams[j].current_value + "'>" + json[i].datastreams[j].current_value + "</span></p>";
                                string += "<p>MAX_VALUE : <span id='maxVal'>" +
json[i].datastreams[j].max_value + "</span></p>";
                                string += "<p>MIN_VALUE : <span id='minVal'>" +
json[i].datastreams[j].min_value + "</span></p>";
                                string += "</div><div class='ButtonSection'>";
                                if (listed == false) {
                                        string += "<input type='button' value='Sonify' id='addToMusic'
onclick='(function(){sonifyAdd(\"" + json[i].id + ":" + json[i].datastreams[j].id + ":" +
json[i].datastreams[j].current_value + ":" + json[i].datastreams[j].max_value + ":" +
json[i].datastreams[j].min_value + "\");})();'>";
                                        string += "<input type='button' value='Remove' style='display:none'
id='removeFromMusic' onclick='(function(){sonifyRemove(\"" + json[i].id + ":" + json[i].datastreams[j].id +
"\");})();'>";
                                }
                                if (listed != false) {
                                        string += "<input type='button' value='Sonify' id='addToMusic'
style='display:none;' onclick='(function(){sonifyAdd(\"" + json[i].id + ":" + json[i].datastreams[j].id + ":" +
json[i].datastreams[j].current_value + "\");})();'>";
                                        string += "<input type='button' value='Remove'
style='display:inline-block;' id='removeFromMusic' onclick='(function(){sonifyRemove(\"" + json[i].id + ":" +
json[i].datastreams[j].id + "\");})();'>";
                                }
                                //id,current_value, max_value, min_value
                                string += "</div>";
                                string += "</div>";
                        }
                        string += "</div>";
                        string += "</ul>";
                        string += "</div>";
                        document.getElementById("feedList").innerHTML = string;
                        document.getElementById("feedList").setAttribute("FeedId", idOfThing);
                }
        }
        if (found == false) {
                document.getElementById("feedList").innerHTML = "Your feed seems to have vanished!";
        }
```

```
	////console.log("thing being calleds");
	displayListUpdate();
}

function getPertinentInfo() {
	if (document.getElementById("feedId") != null) {
		var idOfThing = document.getElementById("feedId").innerHTML;
	}
	for (var i = 0; i < json.length; i++) {
		if (idOfThing == json[i].id) {
			////console.log(json[i]);
			var string = "<div id='dataWrapper'>";
			string += "<h3>" + json[i].title + "</h3><h5 id='uniqueDataFeedDisplayIdentifier'>"
+ json[i].id + "</h5>";
			string += "<p id='creator'>Creator : <a href='" + json[i].creator + "'target='_blank'>"
+ json[i].creator.substring(23, json[i].creator.length) + "</a></p>";
			////console.log();
			if (json[i].description != undefined) {
				string += "<p id='feedDescription'>" + json[i].description + "</p>";
			}
			else if(json[i].description == undefined){
				string += "<p id='feedDescription'></p>";
			}
			string += "<div id='dataStreamDataList'>";
			string += "<ul>";

			for (var j = 0; j < json[i].datastreams.length; j++) {
				var listed = false;
				//chosenFeeds[feedId].push([dataId, currentVal]) ;
				if (chosenFeeds[json[i].id] == undefined) {
					string += "<li><div class='dataWrapping' id='" +
json[i].datastreams[j].id + "'><div class='listItem' >";
				} else if (chosenFeeds[json[i].id] != undefined) {
					var found = false;
					//for loop to see if found.
					for (var zed = 0; zed < chosenFeeds[json[i].id].length; zed++) {
						if (chosenFeeds[json[i].id][zed][0] ==
json[i].datastreams[j].id) {
							found = true;
						}
					}

					if (found == false) {
						string += "<li><div class='dataWrapping' id='" +
json[i].datastreams[j].id + "'><div class='listItem' >";
					}
					if (found != false) {
						listed = true;
```

```
                                                string += "<li><div class='dataWrapping listed' id='" +
json[i].datastreams[j].id + "'><div class='listItem' >";
                                                }
                                        }
                                        string += "<p id='id'>" + json[i].datastreams[j].id + "</p>";
                                        string += "<p>VALUE : <span id='value' data-val='" +
json[i].datastreams[j].current_value + "'>" + json[i].datastreams[j].current_value + "</span></p>";
                                        string += "<p>MAX_VALUE : <span id='maxVal'>" +
json[i].datastreams[j].max_value + "</span></p>";
                                        string += "<p>MIN_VALUE : <span id='minVal'>" +
json[i].datastreams[j].min_value + "</span></p>";
                                        string += "</div><div class='ButtonSection'>";
                                        if (listed == false) {
                                                string += "<input type='button' value='Sonify' id='addToMusic'
onclick='(function(){sonifyAdd(\"" + json[i].id + ":" + json[i].datastreams[j].id + ":" +
json[i].datastreams[j].current_value + ":" + json[i].datastreams[j].max_value + ":" +
json[i].datastreams[j].min_value + "\");})();'>";
                                                string += "<input type='button' value='Remove' style='display:none'
id='removeFromMusic' onclick='(function(){sonifyRemove(\"" + json[i].id + ":" + json[i].datastreams[j].id +
"\");})();'>";
                                        }
                                        if (listed != false) {
                                                string += "<input type='button' value='Sonify' id='addToMusic'
style='display:none;' onclick='(function(){sonifyAdd(\"" + json[i].id + ":" + json[i].datastreams[j].id + ":" +
json[i].datastreams[j].current_value + "\");})();'>";
                                                string += "<input type='button' value='Remove'
style='display:inline-block;' id='removeFromMusic' onclick='(function(){sonifyRemove(\"" + json[i].id + ":" +
json[i].datastreams[j].id + "\");})();'>";
                                        }
                                        //id,current_value, max_value, min_value
                                        string += "</div>";
                                        string += "</div>";
                                }
                                string += "</div>";
                                string += "</ul>";
                                string += "</div>";
                                document.getElementById("feedList").innerHTML = string;
                                document.getElementById("feedList").setAttribute("FeedId", idOfThing);
                        }
                }
}
```

MakeTheCosm/mapping/scripts/transfers.js

```
( function() {"use strict";
                var hashThing = document.URL.substr(document.URL.indexOf('#'), document.URL.length);
                //#blurb, #mailSubscribe, #speakerSubmit
                var hashes = ["", "#soundList", "#soundSelection"];
```

```
		//var hashes = ["","","#soundSelection"]; //multiple blank hashlinks breaks history in this
caes.
		var increment = 0;
		var active = $('.slideshow .slide.active');
		var totalSlides = $('.slide');

		if (hashThing.indexOf("#") != -1) {
			changeScreen(hashThing);
			history.pushState({
				hash : hashThing
			}, "", hashThing);
		}

		$(document).keydown(function(e) {
			if (e.keyCode == 37) {
				backOne();
			}
			if (e.keyCode == 39) {
				nextOne();
			}
		});

		$('#prev-button').click(function() {
			backOne();
		});
		$('#next-button').click(function() {
			nextOne();
		});

		window.addEventListener('popstate', function(event) {
			if (event.state != null) {
				changeScreen(event.state.hash);
			}
			//get event state, calculate distance from current to history using hashes and
increment,
			//simulate clicks off of that.
			//$('#elementid').click();
		});
		function changeScreen(hashToChange) {
			var index;
			for (var i = 0; i < hashes.length; i++) {
				if (hashToChange == hashes[i]) {
					index = i;
				}
			}
			if (index > increment) {
				var nums = index - increment;
				for (var i = 0; i < nums; i++) {
```

```
                                        var next = active.next();
                                        if (next.length) {
                                                active.removeClass('active');
                                                next.addClass('active');
                                                active = next;
                                                if (active[0] != totalSlides[0]) {

document.getElementById("prev-button").className = "";
                                                }
                                                if (active[0] == totalSlides[totalSlides.length - 1]) {

document.getElementById("next-button").className = "hidden";
                                                }
                                        }

                                }
                                increment = index;
                        } else if (increment > index) {
                                var nums = increment - index;
                                for (var i = 0; i < nums; i++) {
                                        var prev = active.prev();
                                        if (prev.length) {
                                                active.removeClass('active');
                                                prev.addClass('active');
                                                active = prev;
                                                if (active[0] == totalSlides[0]) {

document.getElementById("prev-button").className = "hidden";
                                                }
                                                if (active[0] != totalSlides[totalSlides.length - 1]) {

document.getElementById("next-button").className = "";
                                                }
                                        }
                                }
                                increment = index;
                        }
                }

                function nextOne() {
                        var next = active.next();
                        if (next.length) {
                                increment++;
                                historySwapper(increment);
                                active.removeClass('active');
                                next.addClass('active');
                                active = next;
                                if (active[0] != totalSlides[0]) {
```

```
                        document.getElementById("prev-button").className = "";
                    }
                    if (active[0] == totalSlides[totalSlides.length - 1]) {
                        document.getElementById("next-button").className = "hidden";
                    }
                }
            }

            function backOne() {
                var prev = active.prev();
                if (prev.length) {
                    increment--;
                    historySwapper(increment);
                    active.removeClass('active');
                    prev.addClass('active');
                    active = prev;
                    if (active[0] == totalSlides[0]) {
                        document.getElementById("prev-button").className = "hidden";
                    }
                    if (active[0] != totalSlides[totalSlides.length - 1]) {
                        document.getElementById("next-button").className = "";
                    }
                }
            }

            function historySwapper(num) {
                //console.log(num);
                //console.log(hashes[num]);
                //var hashlink = ""+hashes[num].toString()+"";
                var stateObj = {
                    hash : hashes[num]
                };
                if (hashes[num] != "") {
                    history.pushState(stateObj, "", hashes[num]);
                } else if (hashes[num] == "" && document.URL.indexOf('#') != -1) {
                    history.pushState(stateObj, "", document.URL.substr(0,
document.URL.indexOf('#')));
                } else if (hashes[num] == "" && document.URL.indexOf('#') == -1) {
                    history.pushState(stateObj, "", document.URL.substr(0, document.URL));
                }
            }

            if (hashThing.indexOf("#") == -1 || increment == 0) {
                history.pushState({
                    hash : ""
                }, "", hashes[0]);
            }
```

```
}());
```