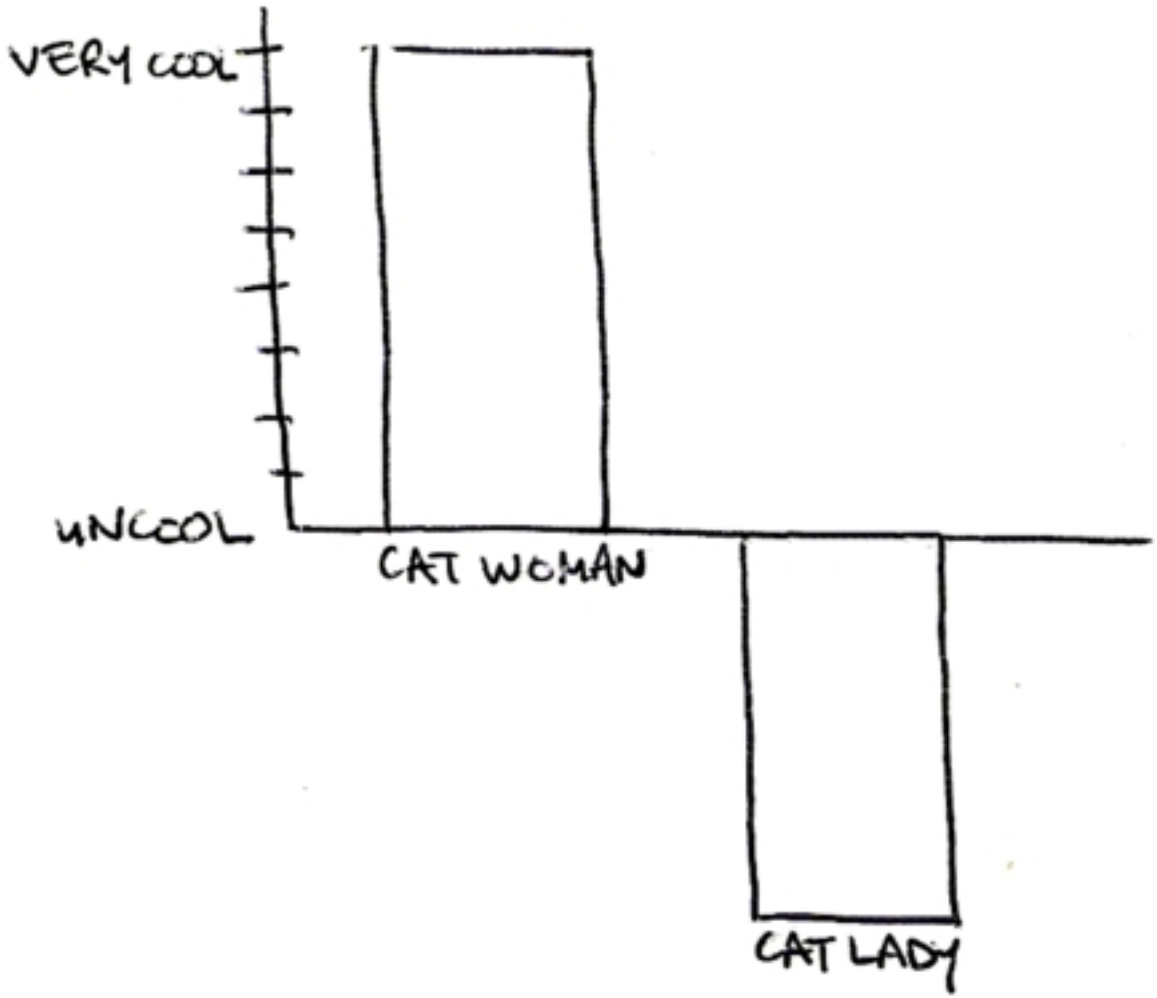
Coder Dojo 2012-02-18

HTML5 Shuffle Puzzle

Uncool

Clean up your rubbish



Creating a simple game

Today we're going to make a simple game using Javascript and HTML5. This game slices up an image, shuffles the pieces about and then the player has to rearrange them.

First off, let's get the image for our game, then we'll have to set up our template page.
To github.com folks....

Go simple...

If you're scared, you already have enough to create a "click Your Own Adventure" game...

Videos...

Just a note (Dublin were on to me...)

The image...

http://github.com/willknott

Look for "CoderDojo-HTML5-CanvasSwapPuzzle"

And find the "CDG.png" image

Image courtesy of Erica

And yes the "CoderDojo" github account will get the files soon.

WARNING

This is a complicated bit of code today.
Pay attention.
We may not complete it today
And we are introducing a LOT of new stuff...

There is no thing as a stupid GENUINE question.
(I said genuine Messers at the back)

If you have any questions... SHOUT THEM OUT

The Template :-)

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8">
    <title>HTML5 Puzzle</title>
    <script src = 'puzzle.js'>
    </script>
</head>
<body onload="init();">
    <canvas id="canvas"></canvas>
</body>
</html>
```

What's happening?

This html page is complete now. The important things are;

- **<script src = 'puzzle.js'>**

This tells us the name of our javascript file. This file is seperate from the HTML.

- **<body onload="init();">**

This onload event tells the computer to load the function "init" when the page opens. This (eventually) puts a picture on our screen.
The question is .....

Thanks to..

http://active.tutsplus.com/tutorials/games/create-an-html5-canvas-tile-swapping-puzzle/
where I stole most of this from
and

Brad Manderscheid from Actionmouse how put the first version together.

What is a function?

Functions are like a machine. They take in something and give back something else.
What other type of machines do we know about?
This meat grinder works like a function.

Examples

- Calculators - turn several numbers into an answer
- Toasters - turns bread into toast!
- A smoothie maker - turns fruit into a drink!
- Diet Coke + Mentos = A Rocket!

Making the game work

Create a new file and save it as puzzle.js, we need to put in all our variables.

```
const PUZZLE_DIFFICULTY = 4;

var  stage;
var  canvas;
var  img;
```

Reminder...

Words have power.

In this case the names of the functions.

If you call Cluck() but meant to call Click() what happens...?

const PUZZLE_DIFFICULTY = 4;

const?

Variables are... variable. They can change

const or constants, don't change.
PUZZLE_DIFFICULTY is just a label.

Why use a label instead of using the value?

Declare more variables

```
var  puzzleWidth;
var  puzzleHeight;
var  pieces;
var  pieceWidth;
var  pieceHeight;
```

And variables we'll use later

```
const PUZZLE_HOVER_TINT = '#FF0000';
var  currentPiece;
var  currentDropPiece;
var  mouse;
```

The first Function

```
function init(){
        img = new Image();
        img.addEventListener('load',onImage,
false);
       img.src = "CDG.png";
     }
```

This loads our image into the HTML page.

addEventListener

addEventListener ties a function with a particular action...

img.addEventListener('load',onImage,false);
'load' is the triggering action

onImage is the function called to action if the action has been triggered

and **false**?
http://www.javascriptkit.com/domref/windowmethods.shtml

Boolean

There is a special variable type called Boolean.
It consists of "true" and "false". No other value.

(In practice, not exactly, but the machine reads "True" or "False")

The third place in an event listened is to do with binding the action, and keep a connection to the following action.
Its false here as we don't need it...

After we load our image, we want to calculate it up into equal sized pieces. You can figure out different sizes later

```
function onImage(e){
    pieceWidth = Math.floor(img.width /
PUZZLE_DIFFICULTY);
    pieceHeight = Math.floor(img.height /
PUZZLE_DIFFICULTY);
    puzzleWidth = pieceWidth * PUZZLE_DIFFICULTY;
    puzzleHeight = pieceHeight * PUZZLE_DIFFICULTY;
    setCanvas();
    initPuzzle();
    }
```

Time for some Math.floor

Math.round = standard round

Math.ceiling = always go up

Math.floor = always go down

Math is the American for Maths

And this does...

We are calculating how to cut the image in to (in our case) 16 equal pieces.

That is 4X4 (more maths, sorry)

Then we set the drawing board called the canvas

Next we want to set up our canvas. We will set up a border so that we can see the edges of the image.

```
// Get element by id,
// on the HTML page, you named the canvas "canvas"
function setCanvas(){
        canvas = document.getElementById('canvas');
        stage = canvas.getContext('2d');
        canvas.width = puzzleWidth;
        canvas.height = puzzleHeight;
        canvas.style.border = "1px solid black";
    }
```

Next, create a function to start the Puzzle

```
function initPuzzle(){
        stage.drawImage(img,
            0, 0, puzzleWidth, puzzleHeight,
            0, 0, puzzleWidth, puzzleHeight);
    }

//Yes we are just drawing the image
//Remember img is the source
// The first set of co-ordinates is the size of the image to use
// and the second set is the size of the canvas to use
```

And when you save and run this..

It "just" displays the image.

Why all the extra bits?

Well we need it later...

And due to running out of time we'll leave it here

Expect more next week