More Javascript
HTML Canvas

- An area you can draw in using standard X,Y coordinates
- Default is to have a pixel independent coordinate system that is 300 wide x 150 high
- “Both the canvas itself and the canvas element have a width and height, they're separate things”
- To draw:
  - get “context”
  - beginPath
    - put stuff onto context
  - closePath
  - fill: define fill and fill
  - lines: define stroke and stroke
Animating the Canvas

- Draw, erase, draw, erase...
- Minor bug... last circle leaves a bit of itself

```javascript
var cnvas = document.getElementById("canvas");
function makeBall() {
  ballcount = ballcount + 1;
  cnvas = document.getElementById("canvas");
  tx = cnvas.clientWidth;
  ty = cnvas.clientHeight;
  ball = new Object();
  ball.radius = Math.random() * (tx * 0.1) + 14;
  ball.x = Math.random() * (tx - 2 * ball.radius) + ball.radius;
  ball.y = Math.random() * (ty - 2 * ball.radius) + ball.radius;
  ball.color = randomColor();
  ball.speed = Math.random() * ball.radius * 0.33 + 1;
  ball.counter = ballcount;
  return ball;
}
function startBall() {
  balls.push(makeBall());
  if (animating == 0) {
    console.log("Interval start");
    animating = 1;
    window.requestAnimationFrame(drawBalls);
  }
}
```

```javascript
function drawBall(ctx, ball) {
  ctx.beginPath();
  ctx.arc(ball.x, ball.y, ball.radius, 0, 2 * Math.PI);
  ctx.closePath();
  ctx.fillStyle = ball.color;
  ctx.fill();
  ctx.stroke();
  ball.x = ball.x + ball.speed;
}
function drawBalls() {
  var tx = cnvas.clientWidth;
  var ty = cnvas.clientHeight;
  var ctx = cnvas.getContext("2d");
  ctx.clearRect(0, 0, cnvas.width, cnvas.height);
  for (i = balls.length - 1; i >= 0; i--) {
    drawBall(ctx, balls[i]);
    // if the ball moves off the screen, remove it from the list
    if ((balls[i].x - balls[i].radius) > cnvas.width) {
      balls.splice(i, 1);
    }
  }
  if (balls.length > 0) {
    window.requestAnimationFrame(drawBalls);
  } else {
    animating = 0;
  }
}
```
function Ball() {
    let tx = cnvas.width;
    let ty = cnvas.height;
    this.radius = Math.random() * (tx * 0.1) + 14;
    this.x = Math.random() * (tx - 2 * this.radius) + this.radius;
    this.y = Math.random() * (ty - 2 * this.radius) + this.radius;
    this.color = randomColor();
    this.speed = Math.random() * this.radius * 0.1 + 0.0001;
    this.counter = ballcount;
    ballcount = ballcount + 1;
}

Ball.prototype.within = function(xx, yy) {
    let cx = this.x;
    let cy = this.y;
    let dxy = Math.sqrt((cx - xx) * (cx - xx) + (cy - yy) * (cy - yy));
    console.log(cx + " " + cy + " " + this.radius + " " + dxy);
    if (dxy < this.radius)
        return this.counter;
    else
        return -1;
};

Ball.prototype.draw = function(ctx) {
    ctx.beginPath();
    ctx.arc(this.x, this.y, this.radius, 0, 2 * Math.PI);
    //console.log(this.counter + " " + this.x + " " + this.y + " " + this.radius);
    ctx.closePath();
    ctx.fillStyle = this.color;
    ctx.fill();
    ctx.stroke();
    this.x = this.x + this.speed;
}

Ball.prototype.offCanvas = function(mx, my) {
    return mx < (this.x - this.radius);
}

• Really ugly code
• Limited to circles
• Not easily extensible
• Lost the onclick function from divs

• Use JS objects!
• XX.prototype adds method to class
  • this.XX get value of XX for the instance of class
• Within method
  • return value could be more useful

• Similar objects for Triangle and N-Gon
• Problem: code assumes all objects have methods within, draw and offcanvas
• Solution: prototype inheritance
Prototype Inheritance

- Create a class “Shape” that has at least stub versions of all methods
- Make Triangle, Ball and Ngon inherit from Shape
- Make each call the Shape “constructor”
• Left N-Gon contains as an exercise
• Now, add a “listener” for mousedown events within the canvas element
• Then get the location of the click and ask each object if it occurred within it.
Last Animating

- A LOT of JS animation not discussed
- A digital Orrery

- file: ssystem.html
Goal:
- allow a page to show the number of times it has been visited
- What data do I need to do this?

Tool:
- PHP Sessions?
- SQL?
  - If SQL table design?
Hit Counters

- file:hitcount0.php

```php
<?php
print updateAndGetCount(100); ?>

<?php
function updateAndGetCount($iidd) {
    $servername = "localhost";
    $username = "gtstudent";
    $password = "";
    $dbname = "count";
    $conn = new mysqli($servername, $username, $password, $dbname);
    if ($conn->connect_error) {
        die("Connection failed: ". $conn->connect_error);
    }
    $sql = "SELECT max(count) as c FROM count WHERE id = $iidd";
    $result = $conn->query($sql);
    $row = $result->fetch_assoc();
    if ($row==null || $row["c"]===null) {
        $visits = 1;
    } else {
        $visits = $row["c"];
        $visits++;
    }
    $sql = "INSERT INTO count (id, count) VALUES ( $iidd, $visits);";
    $result = $conn->query($sql);
    $conn->close();
    return $visits;
} ?>
</body> </html>
```
Improvement to Hit counting

• Should not be the whole page
• Should not just insert into table as that will accumulate a lot of junk
  • file: hitcount.php
    • at least addresses junk problem
  • might be better to use SQL update
    • file: hitcount2.php
Still better HitCount

- Rather than making the whole page a hit count, create a small bit to do the hitcount
- Use javascript fetch to get data from PHP
- The html and the hitcount do not have to come from the same place!
- PHP does not generate HTML, just returns the number of hits

```
<script>
const baseURL = "http://comet.cs.brynmawr.edu/~gtowell/380/Lec10/";
function getHit() {
    const data = { id: 12 }; 
    let fd = new FormData();
    for(var i in data){
        fd.append(i, data[i]);
    }
    fetch(baseURL+"hitcountwidget.php", {
        method: 'POST',
        mode: "cors",
        body: fd,
    }).then(function(response) {
        response.text().then(function(text) {
            console.log($("div.hcc").text() + text);
            $("div.hcc").text("HC:"+text);
        });
    });
}

$(document).ready(function() {
    getHit();
});
</script>

<div id="12" class="hcc"></div>
<div style="height:calc(100% - 50px); margin-top:0px"> ....
```
javascript fetch

- Fetch is a Promise
- First “then” occurs on receiving headers
- In this case body might contain JSON or plain text
  - So examine headers to determine what the body will contain.
    - Invoke a new Promise to get the body of the response and parse appropriately
      - THEN handle the parsed result.

```javascript
fetch(myRequest).then(function(response) {
  const contentType = response.headers.get("content-type");
  if (contentType && contentType.indexOf("application/json") !== -1) {
    return response.json().then(function(json) {
      // process your JSON data further
    });
  } else {
    return response.text().then(function(text) {
      // this is text, do something with it
    });
  }
});
```
More Promising

Two functions in `then` depending on call to `resolve` or `reject` in promise

file: eventloop3.html
Best yet HitCount

file: pagewithhitcount2.html

- Put all of the javascript and supporting CSS into hitcountscript.js and hitcountstyle.css
- Then user only needs to add an element with an attribute hitcountid (along with <link and <script )
- With a little work could put all css into js file
- With a little more work, no jQuery