JavaScript animations and JQuery

cs380

G Towell
Oct 5

Arrays can have holes!

```
<html><head>
        <title>Fibonacci Array</title>
    </head>
    <body>
        <script>
            var arr = new Array();
            arr[1]=1;
            var p1=1;
            var p2=1;
            while (p1<20) {
                let pn = p1 + p2;
                arr[pn] = arr[p1]+arr[p2];
                p1=p2;
                p2=pn;
            }
            var str = arr + "<br>";
            for (index in arr) {
                str = str + index + " " + arr[index] + "<br>";
            }
            str = str + "done<br>";
            for (ii=0; ii<=p2; ii++) {
                if (typeof arr[ii] != 'undefined')
                    str = str + ii + " " + arr[ii] + "<br>";
            }
            document.write(str);
        </script>
    </body></html>
```

file: fibbarr.html

Multiplication Table, v2, part 1

```
<html>
<head>
  <title>Multiplication Table</title>
</head>
<body>
    <table>
      <tr><td>Number of Rows</td><td><input type="text" id="rows" name="cols"></td></tr>
      <tr><td>Number of columns</td><td><input type="text" id="cols" name="cols"></td></tr>
      <tr><td colspan="2"><center><button id="mbutton">Make table</button></center></td></tr>
    </table>
    <table id="mytable">
        <tr><td>Hello</td></tr>
        <tr><td>Goodbye</td></tr>
    </table>
```

Multiplication Table v2, part 2

Set handler for
onclick event
Could have done in html

```
  <script type="text/javascript">

document.getElementById("mbutton").onclick = function() {
    let rows = document.getElementById("rows").value;
    let cols = document.getElementById("cols").value;
    console.log("HHH " + rows + "  " + cols);
    multable(rows, cols)
};

function multable(rows, cols) {
    if(rows == ""  || rows == null)
        rows = 10;
    if(cols== ""  || cols== null)
        cols = 10;
    fillTable(rows, cols);
}
```

Building table one
element at a time and adding
each to DOM.

```
function fillTable(rows, cols)
{
    // first remove all old rows
    var table = document.getElementById("mytable");
    table.innerHTML = "";
    // then create new rows and fill them
    for (rr=0; rr<=rows; rr++) {
        let row = document.createElement("tr");
        table.append(row);
        for (cc=0; cc<=cols; cc++) {
            if (rr==0) {
                let th = document.createElement('th');
                th.innerHTML = ""+(cc==0?"":cc);
                row.appendChild(th);
            } else {
                if (cc==0) {
                    let th = document.createElement('th');
                    th.innerHTML = ""+rr;
                    row.appendChild(th);
                } else {
                    let td = document.createElement("td");
                    td.innerHTML = ""+(rr*cc);
                    row.appendChild(td);
                }
            }
        }
    }
}
```

Multiplication Table, v4

just do it on the server side!

```
<html><head> <title>Multiplication Table</title></head>
<body>
    <form action="mtable.php" type="post">
    <table>
      <tr><td>Number of Rows</td><td><input type="text" id="rows" name="rows"></td>
      <tr><td>Number of columns</td><td><input type="text" id="cols" name="cols"></
      <tr><td colspan="2"><center><button id="mbutton">Make table</button></center>
    </table></form></body></html>
```

```
<?php
function filltable() {
    $rows = $_REQUEST["rows"];
    $cols = $_REQUEST["cols"];
    for ($r=0; $r<=$rows; $r++) {
        echo("<tr>");
        if ($r==0) {
            echo("<th></th>");
        } else {
            echo("<th>$r</th>");
        }
        for ($c=1; $c<=$cols; $c++) {
            if ($r==0) {
                echo("<th>$c</th>");
            } else {
                echo("<td>". ($r*$c) . "</td>");
            }}
        echo("</tr>");
    }}
?>
<html><head><title>Multiplication Table</title></head>
<body>
<table>
<?php filltable(); ?>
</table></body></html>
```

Yes, this this a cheat as we are talking about client-side programming. BUT, it is always correct to ask the question "where should this be done". There is no correct answer.

Meta: both PHP and JS do a lot for html creation. PHP often whole page, JS more often individual elements

Checking User Input

- Best done on client side as it saves a lot of hassle and can be very interactive
- Each input has two fields
  - the form element containing the user input
  - an area for feedback to the user
- UI is checked on every keystroke and on button click
- Note that all this is not wrapped in a form.

```
<body>
    <table id="table1">
      <tr>
        <td>First Name:</td>
        <td><input type="text" id="first" onkeyup="validate();" /></
        <td><div id="errFirst"></div></td>
      </tr>
      <tr>
        <td>Last Name:</td>
        <td><input type="text" id="last" onkeyup="validate();"/></td
        <td><div id="errLast"></div></td>
      </tr>
      <!– not showing email, userid, password and confirm password –
     <tr>
        <td><input type="button" id="create"  value="Create"
             onclick="validate();finalValidate();"/></td>
        <td><div id="errFinal"></div></td>
      </tr>
    </table>
</body>
</html>
```

checkform.html

UI Checking

Javascript

```
var divs = ["errFirst", "errLast", "errEmail", "errUid", "errPassword", "errConfirm"];
var names = ['first', 'last', 'email', 'uid', 'password', 'confirm'];
   function validate() {
     var inputs = new Array();
     inputs[0] = document.getElementById(names[0]).value;
     // …
     var errors = new Array();
     errors[0] = "<span style='color:red'>Please enter your first name!</span>";
     // …
     for (i in inputs) {
       var errMessage = errors[i];
       var div = divs[i];
       if (inputs[i] == "")
           document.getElementById(div).innerHTML = errMessage;
       else if (i==2) {
         var atpos=inputs[i].indexOf("@");
         var dotpos=inputs[i].lastIndexOf(".");
         if (atpos<1 || dotpos<atpos+2 || dotpos+2>=inputs[i].length)
           document.getElementById('errEmail').innerHTML =
                 "<span style='color: red'>Enter a valid email address!</span>";
         else
           document.getElementById(div).innerHTML = "OK!";
       }
       // etc
     }}
```

More UI checking

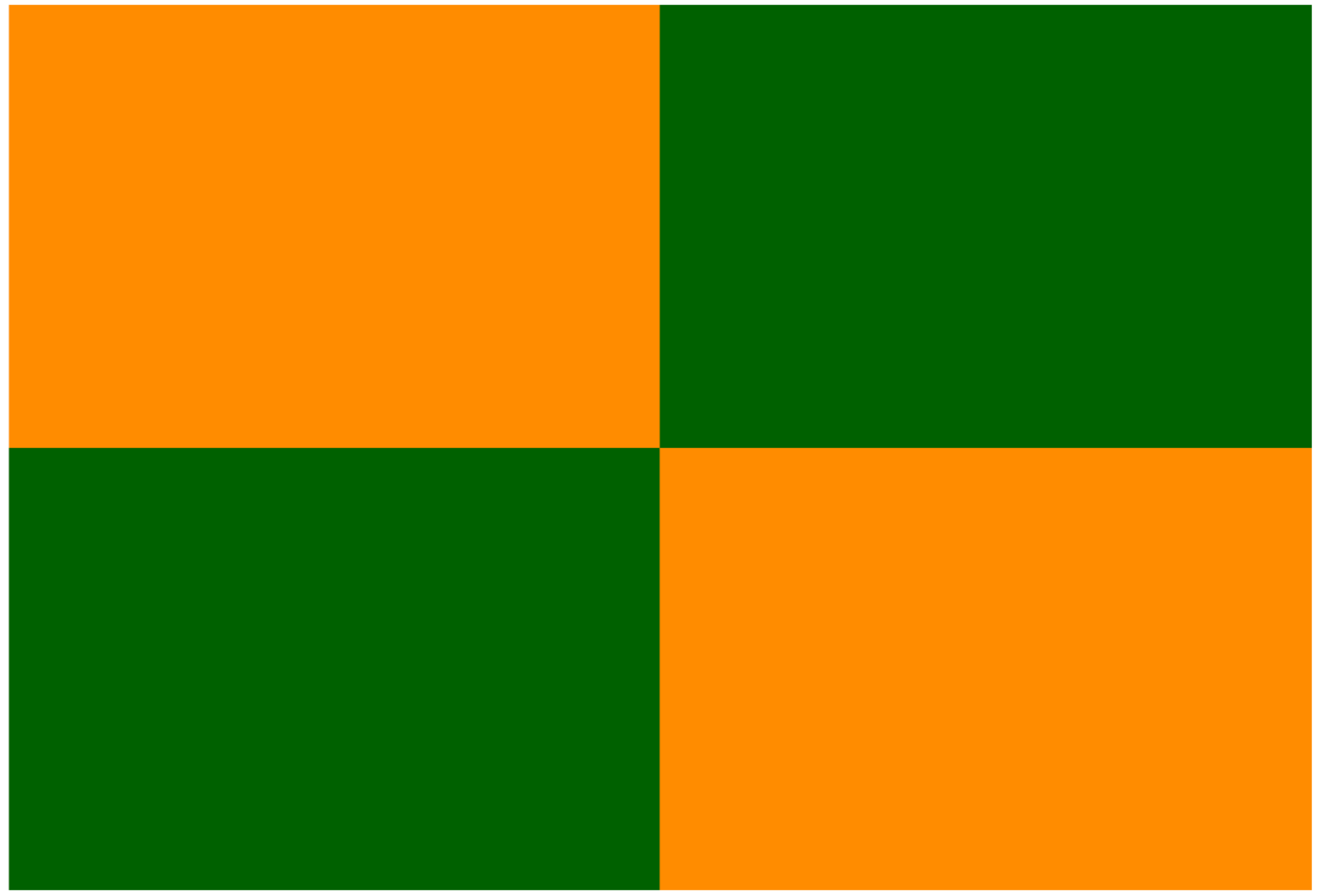
```
function finalValidate()
        {
          var count = 0;
          var params = {};
          for(i=0;i<6;i++)
          {
            var div = divs[i];
            if(document.getElementById(div).innerHTML == "OK!")
                count = count + 1;
            params[names[i]] = document.getElementById(names[i]).value;
          }
          console.log(params);
          if(count == 6) {
                document.getElementById("errFinal").innerHTML =
“All the data you entered is correct!!!";
                post("former.php", params, 'post');
          }
        }
```

```
function post(path, params, method='post') {
            const form = document.createElement('form');
            form.method = method;
            form.action = path;
            for (key in params) {
            if (params.hasOwnProperty(key)) {
                hiddenField = document.createElement('input');
                hiddenField.type = 'hidden';
                hiddenField.name = key;
                hiddenField.value = params[key];
                form.appendChild(hiddenField);
                }
            }
            document.body.appendChild(form);
            form.submit();
            }
```

Checkerboard

from lecture 1

- Remember this?
- Critiques?
  - Only 2x2
  - fixed colors
- Solution
  - forms and javascript!



Recall the HTML

```
<html>
    <style>
        .container { margin-left:5%; width:90%; height:50%; display: flex; }
        .shape {
            margin:0pt; width:50%; height:100%; display: inline-block; padding: 0pt;
        }
        .color1 { background-color:#03335f; }
        .color2 { background-color:#f3b720; }
    </style>
    <body>
        <div class="container">
            <div class="shape color1"></div>
            <div class="shape color2"></div>
        </div>
        <div class="container">
            <div class="shape color2"></div>
            <div class="shape color1"></div>
        </div>
    </body>
</html>
```

Javascript checkerboard

Initial HTML

divs are absolute positions, so they are on top of each other. z-index controls who is on top

```
    <body>
        <div id="thediv2" style="background-color: red; margin-top:20%; margin-left:10%; width:80%;
height:80%; display: flex; z-index: -1; position: absolute;">
        </div>
        <div id="thediv" style="margin-top:1%; margin-left:1%; width:98%; height:98%; display:
flex; z-index: 1; position:absolute">
            <table>
                <tr><td>Grid size (1-100)</td><td><input type="text" id="first"
onkeyup="checker();" /></td></tr>
                <tr><td>Color 1</td><td><input type="text" id="color1"/></td></tr>
                <tr><td>Color 2</td><td><input type="text" id="color2"/></td></tr>
                <tr><td colspan="2" align="center"><button id="abutton" onclick="setColors();">Set
Colors</button></td></tr>
            </table>
        </div>

    </body>
```

checkerboarding changes on every keystroke

Javascript checkerboard

```
var color1="pink";
var color2="white";
function setColors() {
    if (document.getElementById("color1").value.length > 0)
        color1=document.getElementById("color1").value;
    if (document.getElementById("color2").value.length > 0)
        color2=document.getElementById("color2").value;
    checker();
}
/**
  * takes a number (x) and a precision (digits) and
  * returns a representation with exactly that number of sig digits
  */
function precise(x, digits) {
    return Number.parseFloat(x).toPrecision(digits);
}
```

Javascript checkerboard

```
        function checker() {
             var val = document.getElementById("first").value;
             if (isNaN(val) || (val<0) || (val>100)) return;
             var div = document.getElementById("thediv");
             div.innerHTML="";
             var newHTML = "";
             pct = 100/val;
             for (j=0; j<val; j++) {
                 newHTML=newHTML+"<div style=\"height:100%; width:"+precise(pct,4)+"%;\">";
                 for (k=0; k<val; k++) {
                     color=color1;
                     if ((j+k)%2==0)
                         color=color2;
                     newHTML=newHTML+"<div style=\"background-color:"+color+"; width:100%;
height:"+precise(pct,4)+"%;\"></div>"
                 }
                 newHTML=newHTML+"</div>";
             }
             console.log(newHTML);
             div.innerHTML=newHTML;
        }
```

Idea: build exactly the html that I wrote by hand to make a checkerboard, then put that in the div. Alternately I could have made each element and added the element to the correct spot in the DOM.

JQuery: “Write less, do more”

- 2 ways to “do more”
  - Use their Javascript extensions
    - <script src=“JQ/jquery-1.9.1.min.js”></script> (Current is 3.5.1)
  - Use their interface widgets
    - JQuery mobile
      - course homepage
        - work with demo library
          - <script src=“JQ/jquery.mobile-1.4.5.min.js"></script>
          - <link rel="stylesheet" href=“JQ/jquery.mobile-1.4.5.min.css">

Javascript and JQuery

Selection

```
<!DOCTYPE html>
<html>
<body>
<p>An unordered list:</p>
<ul>
  <li>Coffee</li>
  <li>Tea</li>
  <li>Milk</li>
</ul>
<button onclick="myFunction()">Try it</button>
<script>
function myFunction() {
    var x = document.getElementsByTagName("LI");
    for (i = 0; i < x.length; i++) {
        x[i].style.backgroundColor = "red";
    }
}
</script>
</body>
</html>
```

file:d1.html

```
<!DOCTYPE html>
<html>
<body>
<script src="../JQ/jquery-1.9.1.min.js"></script>

<p>An unordered list:</p>
<ul>
  <li>Coffee</li>
  <li>Tea</li>
  <li>Milk</li>
</ul>
<button onclick='$("li").css("background-color", “red")'>
Try it</button>
</body>
</html>
```

file: d2.html, d2.1.html, d3.html

Could this be improved?

More JQuery

```
<!DOCTYPE html>
<html>
<body>
<script src="../JQ/jquery-1.9.1.min.js"></script>

<p>An unordered list</p>
<ul>
  <li id="coffee" onclick="$('li').css('background-color', 'transparent');$('#coffee').css('background-color', 
'brown');">Coffee</li>
  <li id="tea" onclick="$('li').css('background-color', 'transparent');$('#tea').css('background-color', 'gold');">Tea</
li>
  <li id="milk" onclick="$('li').css('background-color', 'transparent');$('#milk').css('background-color', 
'blue');">Milk</li>
</ul>
```

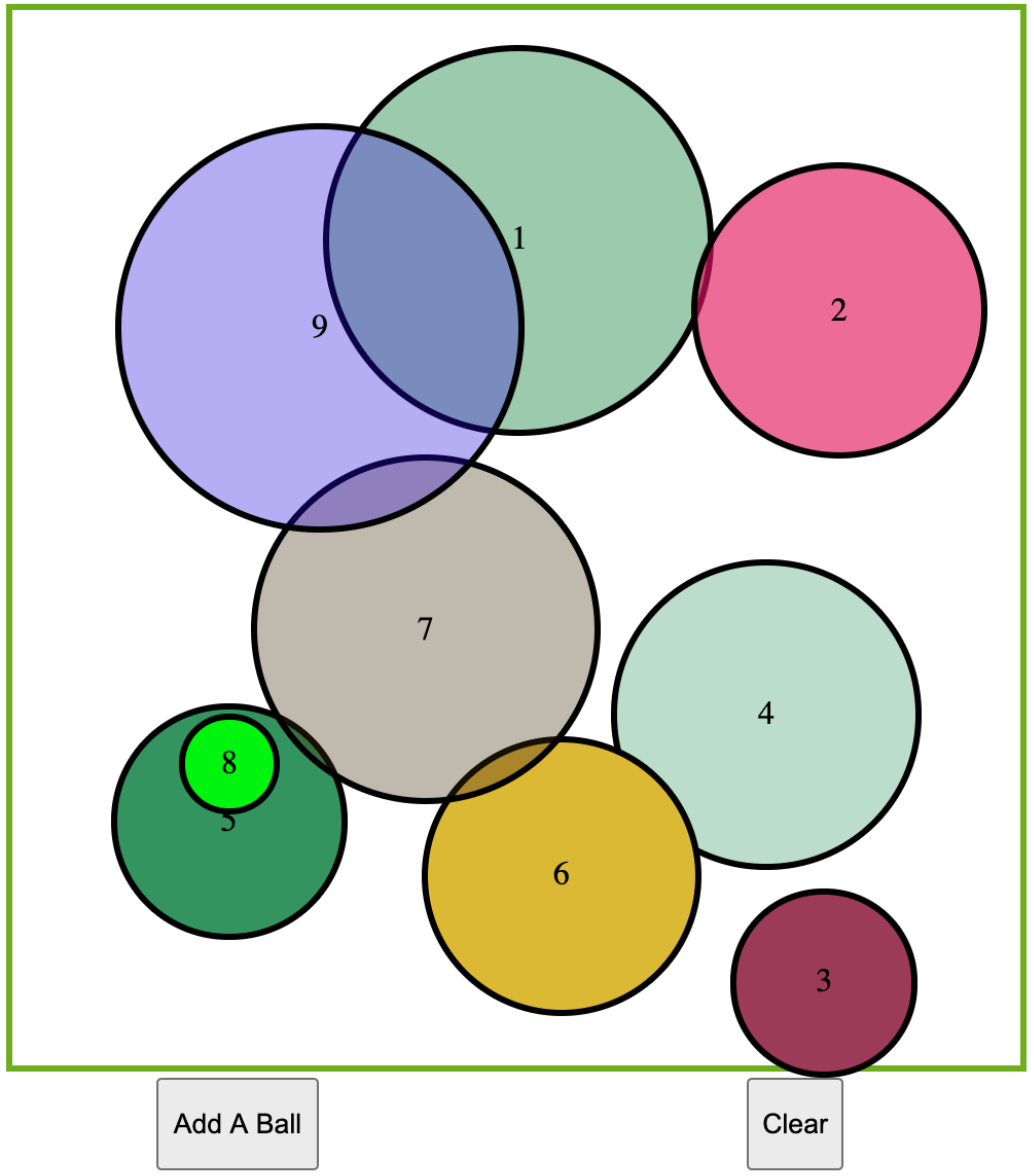
d4.html

```
<script>
    function colorIt(id, color) {
        $("#"+id).css("background-color", color);
        $("li").each(function(index) {
            console.log("ID: " + $(this).attr('id'));
            if (id==$(this).attr('id')) {
                console.log("Have self"  + id);
            } else {
                $(this).css("background-color", "transparent");
            }
        });
    }
</script></body></html>
```

JQuery warnings

- avoid mixing with regular javascript objects
    - var x = document.getElementById(“id”)
      x.css(“background-color”:”red”);
    - Convert regular javascript object to JQuery using $() (but why?)
        - var x = document.getElementById(“id”)
          $(x).css(“background-color”:”red”);
- You can, but should avoid, searching entire DOM for objects. Better to get known enclosing item then search there
    - $("ul").find("li").css("background-color", "red")
    - Also, use variables to hold objects rather than re-search.
    - (Not important when docs are small)

Screen Drawing



b2.html

“Drawing” with divs

idea: make a div and put it where you want it

```
<html>
    <head>
        <script src="../JQ/jquery-1.9.1.min.js"></script>
        <style>
            .maind {
                width: 90%;
                height: 90%;
                margin-left: 50px;
                margin: auto;
                border: 3px solid #73AD21;
            }
            .balld {
                position: absolute;
                text-align: center;
                border: 3px solid black;
            }
            .bttn {
                height:45px
            }
        </style>
    </head>
```

DwD, part 2

The Layout

```
    <body>
        <div id="mdiv" class="maind"></div>
        <table width="100%">
        <tr><td width="50%"><center><button class="bttn" onclick="makeBall()">Add A Ball</button></center></td>
            <td><center><button class="bttn" onclick="clearBalls()">Clear</button></center></center></td>
        </tr>
        </table>
```

DwD, part 3

Javascript (using JQuery)

```
<script>
    //The number of balls created.
    var counter = 0;

    /**
     * Create a random color.  Actually this returns a string
     * which can be evaluated into a random color
     */
    function randomColor() {
        return (
            "rgba(" +
                Math.round(Math.random() * 250) + "," +
                Math.round(Math.random() * 250) + "," +
                Math.round(Math.random() * 250) + "," +
                Math.ceil(Math.random() * 10) / 3 + ")"
            );
        }
    /**
     * Clear all of the balls
     */
     function clearBalls() {
        $(".balld").remove();
    }
```

```
        function makeBall() {
                counter=counter+1;
                cnvas = $("#mdiv"); // get the place where the ball will
                tx = cnvas.width();
                ty = cnvas.height();
                radius = 14 + Math.random() * (0.4*(tx<ty ? tx : ty)); //
                x = Math.random() * (tx-radius); // ball location
                y = Math.random() * (ty-radius);
                // make the ball
                jelem = $('<div>'+counter+"</div>"); //document.createEle
                jelem.addClass("balld");
                jelem.css( {
                    'line-height':radius+'px',
                    'margin-left': x+'px',
                    'margin-top':y+'px',
                    "height": radius+'px',
                    'width':radius+'px',
                    'border-radius':radius+'px',
                    'background-color': randomColor() });
                jelem.hover(function(){  //mouseover and mouseout
                    $(this).css('border', '3px solid yellow');
                    }, function(){
                    $(this).css('border', '3px solid black');
                });
                //put the ball into the target div
                cnvas.append(jelem);
            }
```

Drawing with Canvas

- canvas is an html element that you can literally draw on.
- Just doing circles so everything here could be done with divs
- Diagonal lines, etc not so much
- canvas and jquery do not talk well so using base javascript for canvas

```
   <head>
        <style>
            .maind {
                border: 3px solid #73AD21;
            }
        </style>
    </head>
    <body>
        <script src="../JQ/jquery-1.9.1.min.js"></script>
        <canvas id="canvas" class="maind"></canvas>
        <button onclick="startBall()">New Ball</button>
```

DwC, Javascript pt 1

```
var ballcount=0;
var balls = [];
var animating=0;
$(document).ready(
    function() {
        console.log("A");
        console.log("b");
        ww = Math.floor(0.9*window.innerWidth);
        wh= Math.floor(0.9*window.innerHeight);
        cnvas = document.getElementById("canvas");
        cnvas.width=ww;;
        cnvas.height=wh;
    }
);
function randomColor() {
    return (
        "rgba(" +
            Math.round(Math.random() * 250) + "," +
            Math.round(Math.random() * 250) + "," +
            Math.round(Math.random() * 250) + "," +
            Math.ceil(Math.random() * 10) / 3 + ")"
        );
    }
```

DwC, javascript part 2

```
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;
    //drawBall(ball);
}
function startBall() {
    balls.push(makeBall());
    if (animating==0) {
        console.log("Interval start");
        animating=1;
        window.requestAnimationFrame(drawBalls);
    }
}
```

DwC, Javascript part 3

```
function drawBall(ctx, ball) {
    ctx.beginPath();
    ctx.arc(ball.x, ball.y, ball.radius, 0, 2 * Math.PI);
        console.log(ball.counter + " " + ball.x + " " + ball.y + " " + ball.radius);
        ctx.fillStyle = ball.color;
        ctx.fill();
        ctx.stroke();
        ball.x=ball.x+ball.speed;

}
function drawBalls() {
    cnvas = document.getElementById("canvas");
    var tx = cnvas.clientWidth;
    var ty = cnvas.clientHeight;
    var ctx = cnvas.getContext("2d");
    ctx.clearRect(0, 0, tx, ty);
    for (i=balls.length-1; i>=0; i--) {
        drawBall(ctx, balls[i]);
        if ((balls[i].x-balls[i].radius) > tx) {
            balls.splice(i,1);
        }
    }
    if (balls.length>0)
        window.requestAnimationFrame(drawBalls);
    else
        animating=0;
}
```

Clicks in a Canvas

```
$(document).ready(
    function() {
        ww = Math.floor(0.9*window.innerWidth);
        wh= Math.floor(0.9*window.innerHeight);
        cnvas = document.getElementById("canvas");
        cnvas.width=ww;;
        cnvas.height=wh;

        cnvas.addEventListener("mousedown", function(e)
        {
            getMousePosition(cnvas, e);
        });
    }
);
function getMousePosition(canvas, event) {
    let rect = canvas.getBoundingClientRect();
    let x = event.clientX - rect.left;
    let y = event.clientY - rect.top;
     console.log("Coordinate x: " + x,  "Coordinate y: " + y);
}
```

Scope in Javascript, when do variables exist?

DwD, adding animation

Adjustments to CSS and html layout

```
   .bttn {
        height:45px;
        width:50%
    }
    .telem {
        width:33%;
        text-align: center;
    }
```

```
    <body>
        <div id="mdiv" class="maind"></div>
        <table width="100%">
        <tr><td class="telem"><button class="bttn"
onclick="addBall()">Add A Ball</button></td>
            <td class="telem"><button class="bttn"
onclick="moveOneStep()">Move</button></td>
            <td class="telem"><button class="bttn"
onclick="clearBalls()">Clear</button></td>
        </tr>
        </table>
```

file:b2anim.html, b2animu.html

DwD, adding animation

Javascript changes

```javascript
   // unchanged above here
    cnvas.append(jelem);
    // new .. create and fill an object for each ball
    ball = new Object();
    ball.element = jelem;
    ball.addx = Math.random()*20*(Math.random()>0.5?1:-1);
    ball.addy = Math.random()*20*(Math.random()>0.5?1:-1);
    ball.radius = radius;
    ball.xloc=x;
    ball.yloc=y;
    return ball;
}

var balls = [];
function addBall() {
    balls.push(makeBall());
}
```

```javascript
function moveOneStep() {
    for (ball of balls) {
        let cnvas = $("#mdiv"); // get the place where th
        let tx = cnvas.width();
        let ty = cnvas.height();
        ball.xloc += ball.addx;
        ball.yloc += ball.addy;
        if (ball.xloc >= (tx-ball.radius)) {
            ball.xloc=tx-ball.radius;
            ball.addx=-ball.addx;
        }
        if (0>ball.xloc) {
            ball.xloc=0;
            ball.addx = -ball.addx;
        }
        if (ball.yloc >= (ty-ball.radius)) {
            ball.yloc = ty-ball.radius;
            ball.addy = -ball.addy;
        }
        if (0 > ball.yloc) {
            ball.yloc = 0;
            ball.addy = -ball.addy;
        }
        ball.element.css({   'margin-left': ball.xloc+'px
        'margin-top':ball.yloc+'px',});
    }
}
```