Positioning divs in a circle using JavaScript

Asked 10 years, 1 month ago Modified 19 days ago Viewed 21k times

25

I am trying to position 15 `div` elements evenly in a circle with a radius of `150px`. I'm using the following code, which seems to give an oddly eccentric ellipse that overlaps.

Fiddle

```
// Hold a global reference to the div#main element.  Initially assign it ...
somewhere useful :)
var main = document.getElementById('main');
var circleArray = [];

// Move a circle based on the distance of the approaching mouse
var moveCircle = function(circle, dx, dy) {

};

// Look at all the circle elements, and figure out if any of them have to move.
var checkMove = function() {

};
var setup = function() {
  for (var i = 0; i < 15; i++) {
    //create element, add it to the array, and assign it's coordinates
trigonometrically.
    //Then add it to the "main" div
    var circle = document.createElement('div');
    circle.className = 'circle number' + i;
    circleArray.push(circle);
    circleArray[i].posx = Math.round((150 * Math.cos(i * (2 * Math.PI / 15))))
+ 'px';
    circleArray[i].posy = Math.round((150 * Math.sin(i * (2 * Math.PI / 15))))
+ 'px';
    circleArray[i].style.position = "relative";
    circleArray[i].style.top = circleArray[i].posy;
    circleArray[i].style.left = circleArray[i].posx;
    main.appendChild(circleArray[i]);
  }
};
setup();
window.addEventListener('load', function() {

});
```

```
div {
  box-sizing: border-box;
}
div#main {
  position: absolute;
  left: 50%;
  top: 50%;
```

```
}
div.circle {
  position: absolute;
  width: 20px;
  height: 20px;
  border: 2px solid black;
  border-radius: 10px;
  -webkit-border-radius: 10px;
  -moz-border-radius: 10px;
}
```

```
<div id="main"></div>
```

Run code snippet Expand snippet

Any suggestions as to what I might be doing wrong?

javascript html css

Share
Improve this question
Follow

edited Jan 7, 2015 at 14:56

Weafs.py
23k ● 9 ● 57 ● 79

asked Oct 28, 2014 at 2:03

Araymer
1,525 ● 1 ● 12 ● 16

5 Answers

Sorted by: Highest score (default)

82

First of all, the equation for a co-ordinate on a circle is simply:

```
(x, y) = (r * cos(θ), r * sin(θ))
```

where, `r` is the radius of a circle and `θ` is the angle in radians.

The reason why your code is creating an eccentric ellipse is because when you assign the `.top` and `.left` CSS values, you are not considering that it will actually take the top-left corner as its reference. I've fixed your code and now it creates a perfect circle.

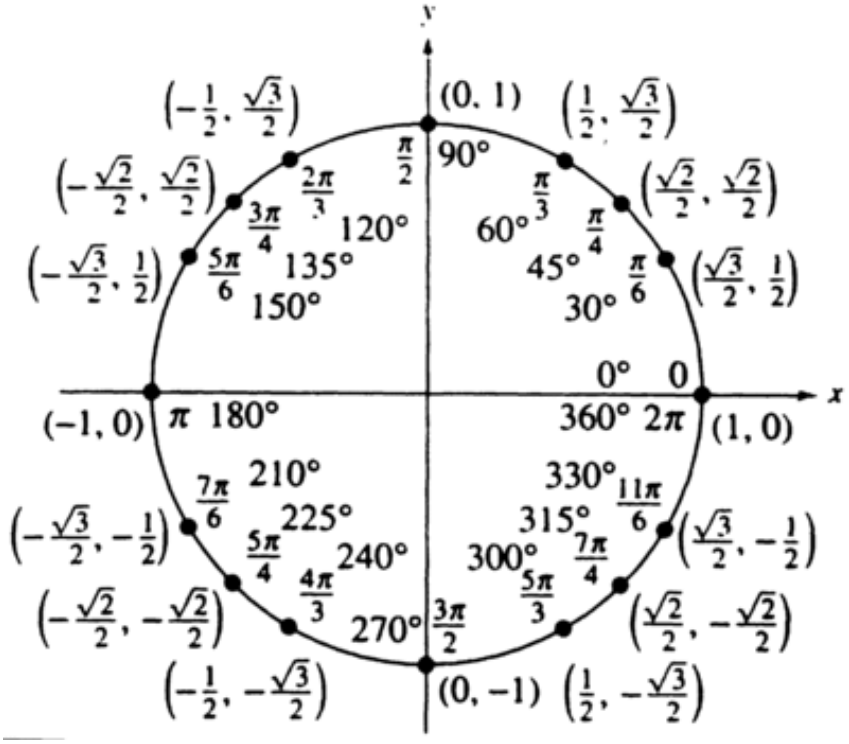
Changes made to your code:

1. Added an array `theta` that holds all the angles.

```
var theta = [0, Math.PI / 6, Math.PI / 4, Math.PI / 3, Math.PI / 2, 2 *
(Math.PI / 3), 3 * (Math.PI / 4), 5 * (Math.PI / 6), Math.PI, 7 * (Math.PI /
```

```
6), 5 * (Math.PI / 4), 4 * (Math.PI / 3), 3 * (Math.PI / 2), 5 * (Math.PI / 3)
7 * (Math.PI / 4), 11 * (Math.PI / 6)];
```

The image below shows all the angles I've used.



2. Added an array `colors` that holds different colors.

```
var colors = ['red', 'green', 'purple', 'black', 'orange', 'yellow', 'maroon',
'grey', 'lightblue', 'tomato', 'pink', 'maroon', 'cyan', 'magenta', 'blue',
'chocolate', 'DarkSlateBlue'];
```

3. Made changes to your trigonometric equations.

```
circleArray[i].posx = Math.round(radius * (Math.cos(theta[i]))) + 'px';
circleArray[i].posy = Math.round(radius * (Math.sin(theta[i]))) + 'px';
```

4. Changed the way `.top` and `.left` are assigned.

```
circleArray[i].style.top = ((mainHeight / 2) -
parseInt(circleArray[i].posy.slice(0, -2))) + 'px';
circleArray[i].style.left = ((mainHeight / 2) +
parseInt(circleArray[i].posx.slice(0, -2))) + 'px';
```

where `mainHeight` is the height of the `#main` `div`.

---

[1] 16 `div`s

Demo on Fiddle

▶ Show code snippet

---

[2] 15 `div`s Positioned Evenly

Demo on Fiddle

Show code snippet

---

[3] Dynamically Position any number of `div`s on an Ellipse/Circle

The equation for a co-ordinate on an ellipse is:

$(x, y) = (r_x * \cos(\theta), r_y * \sin(\theta))$

where, r_x is the radius along X-axis and r_y is the radius along Y-axis.

In this case, the function `generate(n, rx, ry, id)` takes four arguments, where `n` is the number of `div`s, `rx` and `ry` are the radii along the X and Y-axis respectively and finally `id` is the `id` of the `div` that you want to append your elliptically arranged `div`s in.

Demo on Fiddle

Show code snippet

Edit[9th December 2015]:

Here's a more flexible version with start offset, clock wise and anti-clock wise functionality.

```
/*
Usage: Position.ellipse(n, rx, ry, so, wh, idd, cls, cw);

where n = number of divs,
      rx = radius along X-axis,
      ry = radius along Y-axis,
      so = startOffset,
      wh = width/height of divs,
      idd = id of main div(ellipse),
      cls = className of divs;
      cw = clockwise(true/false)
*/

var Position = {
  ellipse: function(n, rx, ry, so, wh, idd, cls, cw) {
    var m = document.createElement('div'),
```

```
      ss = document.styleSheets;
    ss[0].insertRule('#' + idd + ' { position: absolute; left: 50%; top: 50%;
transform: translate(-50%, -50%); border-radius: 50%; box-shadow: inset 0 0 ' +
wh + 'px ' + wh / 4 + 'px black; background: rgba(0, 0, 0, 0.2); width: ' +
String((rx * 2) + wh) + 'px; height: ' + String((ry * 2) + wh) + 'px; }', 1);
    ss[0].insertRule('.' + cls + '{ position: absolute; background: black;
color: papayawhip; text-align: center; font-family: "Open Sans Condensed",
sans-serif; border-radius: 50%; transition: transform 0.2s ease; width: ' + wh
+ 'px; height: ' + wh + 'px; line-height: ' + wh + 'px;}', 1);
    ss[0].insertRule('.' + cls + ':hover { transform: scale(1.2); cursor:
pointer; background: rgba(0, 0, 0, 0.8); }', 1);
    m.id = idd;
    for (var i = 0; i < n; i++) {
      var c = document.createElement('div');
      c.className = cls;
      c.innerHTML = i + 1;
      c.style.top = String(ry + -ry * Math.cos((360 / n / 180) * (i + so) *
Math.PI)) + 'px';
      c.style.left = String(rx + rx * (cw ? Math.sin((360 / n / 180) * (i + so)
* Math.PI) : -Math.sin((360 / n / 180) * (i + so) * Math.PI))) + 'px';
      m.appendChild(c);
    }
    document.body.appendChild(m);
  }
}

Position.ellipse(20, 150, 150, 0, 42, 'main', 'circle', true);
```

```
@import url(http://fonts.googleapis.com/css?family=Open+Sans+Condensed:300);
 body {
  margin: 0 auto;
  background: rgb(198, 193, 173);
}
```

Run code snippet Expand snippet

Share Improve this answer Follow

edited Feb 27, 2018 at 22:50

answered Oct 28, 2014 at 4:44

Weafs.py
23k ● 9 ● 57 ● 79

1 This is pretty complete, thanks! I am curious about your use of the slice method. I think I misunderstood what it's functionality is (I thought it related more to strings?) – Araymer Oct 28, 2014 at 22:47

`.slice(0, -2)` simply extracts, For example `500` from `500px`. You can use `.slice()` method on arrays as well. – Weafs.py Oct 29, 2014 at 0:38

2 Great post, super informative ! Forked Fiddle [5] with random circle numbers and diameters at jsfiddle.net/Lq7tkxj3 – Antoine Dec 9, 2015 at 13:35

can we do the same thing with elliptical shape? – Varada Jun 30, 2016 at 12:08

1 This doesn't look correct, see the circle here: jsfiddle.net/s3pgwcth The small circles are not aligned correctly outside of the big circle. The x and y need to take in to account the size of the

small circle, if we subtract 10 from x and add 10 to y we get a perfect position: jsfiddle.net/cL68os91 – AntonB May 21, 2021 at 15:19

18

Other approach without JS

chipChocolate.py's anser is pretty complete but there is an other way to achieve your aim. It is simpler and doesn't require JS.

The point is to think **"circle" and rotation** rather than rely on `[x,y]` coordinates :

You need to nest all the elements and apply a rotation to them. As they are nested the `n + 1` element will rotate according to it's direct parent's rotation. Here is a **DEMO** :

```
.circle, .circle div {
    width:24px; height:300px;
    position:absolute;
    left:50%; top:50px;
}
.circle:before, .circle div:before {
    content:'';
    display:block;
    width:20px; height:20px;
    border: 2px solid black;
    border-radius: 100%;
}
.circle div {
    top:0; left:0;
    -webkit-transform : rotate(24deg);
    -ms-transform : rotate(24deg);
    transform : rotate(24deg);
}
```

```
<div class="circle">
    <div><div><div><div><div><div><div><div><div><div><div><div><div><div><div><div>
    </div></div></div></div></div></div></div></div></div></div></div></div></div></div>
</div></div></div>
</div>
```

Run code snippet Expand snippet

The diameter of the circle is controled by the height of the elements (in the demo `height:300px`) you can make that a percentage to make the circle responsive (see below).

The rotation must be set according to the number of elements you want around the circle. In the demo 15 elements so `rotation = 360 / 15 = 24deg`.

If you have a dynamic number of elements, you may use JS to add them and to calculate the rotation angle needed.

Responsive example

DEMO

```
.circle{
    position:relative;
    width:5%;padding-bottom:50%;
    margin-left:47.5%;
}
.circle div {
    position:absolute;
    top:0; left:0;
    width:100%; height:100%;
    -webkit-transform : rotate(24deg);
    -ms-transform : rotate(24deg);
    transform : rotate(24deg);
}
.circle:before, .circle div:before {
    content:'';
    position:absolute;
    top:0; left:0;
    width:100%; padding-bottom:100%;
    border-radius: 100%;
    border: 2px solid teal;
    background:gold;
}
```

```
<div class="circle">
    <div><div><div><div><div><div><div><div><div><div><div><div><div><div><div><div>
    </div></div></div></div></div></div></div></div></div></div></div></div></div></div>
</div></div></div>
</div>
```

Run code snippet Expand snippet

Share Improve this answer

Follow

edited Oct 28, 2014 at 10:14

answered Oct 28, 2014 at 9:31

web-tiki

104k ● 33 ● 224 ● 257

I have to position them within javascript because I'm going to write code to manipulate the positions based on mousemove and need the script to be aware of their positions to begin with. – Araymer Oct 28, 2014 at 22:48

3 ok, I'll leave this answer in case someone needs the same thing without JS. – web-tiki Oct 29, 2014 at 8:23

1 Great for me too. New native CSS capabilities may reduce the need to use complex javascript, so, when possible I prefer to stick to native instead of javascript. Nevertheless, huge amount of knowledge in the javascript answer. – Paulo Janeiro Jun 13, 2016 at 18:41

2

Yet another solution, based on ideas from other solutions I've seen

http://jsfiddle.net/0hr1n7a2/6/

```
(function() {
  var radians, radius;

  radius = 150;

  var totalItems = 48
  var item = 0;
  function positionTarget()
  {
    var x, y, angle = 0, step = (2*Math.PI) / totalItems;
    var width = $('#container').width()/2;
    var height = $('#container').height()/2;
    var itemW = 20, itemH = 2;
    var deg = 0;
 while(item <= totalItems)
 {
  x = Math.round(width + radius * Math.cos(angle) - itemW/2);
  y = Math.round(height + radius * Math.sin(angle) - itemH/2);
        //console.log(x + "," + y);

  $('#container').append('<div id="'+ item +'"/>')
  $('div#'+item).css('position', 'absolute')
        .css('width', itemW+'px').css('height', itemH+'px')
        .css('left', x+'px').css('top', y+'px')
        .css('background-color', 'blue')
        .css('transform-origin', x+'px' -y+'px')
        .css('transform', 'rotate('+ deg +'deg)')
        .css('border', 'solid 1px #000');

        angle += step;
        ++item;
        deg += 360/totalItems;

        //console.log(deg)
    }
  }

  $('#theButton').on('click', function()
  {
        positionTarget();
  })

})();
```

```
#container { width: 600px; height: 600px; border: 1px solid #000; position:
relative; }
```

```
<script src="https://ajax.googleapis.com/ajax/libs/jquery/2.1.1/jquery.min.js">
</script>
<input type="button" id="theButton" value="Draw">
<div id="container">
</div>
```

Run code snippet Expand snippet

Share Improve this answer Follow

answered Apr 12, 2015 at 21:07

Michael Worku

31 ● 2

0

1. Set the position to "absolute". This will allow "top" and "left" to position the divs from (0, 0). Using "relative" will position the divs from where they would normally be laid out.
2. Change the center point of your circle from (0, 0) to something else, like (250, 250).

```
circleArray[i].posx = 250 + Math.round((150*Math.cos(i*(2*Math.PI/15)))) +
'px';
circleArray[i].posy = 250 + Math.round((150*Math.sin(i*(2*Math.PI/15)))) +
'px';
circleArray[i].style.position = "absolute";
```

Share Improve this answer Follow

edited Oct 28, 2014 at 2:26

arserbin3

6,150 ● 8 ● 38 ● 53

answered Oct 28, 2014 at 2:23

Chris

101 ● 1 ● 7

I want them to be positioned relatively from where they'd normally be positioned (which is the center of the page as a child of the "main" div, so they're all circling the center of the page, basically) – Araymer Oct 28, 2014 at 2:52

0

Here's another take at it learning from the existing answers, 10 years later.

FIDDLE

```
<div class="circle" id="main"></div>
```

```
// cache
let c = document.querySelector("#main");

// constants
const pi = Math.PI;
let R = 180; // let, because we can dynamically change this
```

```
let count = 19;
let DELTA = 360 / count;
let TRANSPOSE_OFFSET = R + 50;

// helper functions
const pxfy = (n) => `${n}px`;
const radians = (theta) => (theta * pi) / 180;
const make = (e) => document.createElement(e);
const r_cosA = (r, a) => Math.cos(a) * r;
const r_sinA = (r, a) => Math.sin(a) * r;
const transpose = (p, delta) => {
  // our circle is drawn at the origin
  // transpose its center to where you like
  let x = p.x + delta.x;
  let y = p.y + delta.y;
  return { x, y };
};

const get_point = (r, a) => {
  // {x,y} = {r * cos(a), r * sin(a)}
  let x = r_cosA(r, a);
  let y = r_sinA(r, a);
  // where do you want to move the center?
  let Dx = TRANSPOSE_OFFSET + 10;
  let Dy = TRANSPOSE_OFFSET;
  return transpose({ x, y }, { x: Dx, y: Dy });
};

// THE CODE TO DRAW CIRCLES
const setup = () => {
  // additional decorations
  let text = "This is JavaScript";
  text = text.padStart(count, '*');
  let text_split = text.split('');
  // ---

  // reference range array
  let range = [...Array(count).keys()];

  range.forEach((i) => {
    let circle = make("div");
    circle.id = `c_${i}`;
    circle.textContent = `${i + 1}`;

    // additional decorations
    let elem = make('div');
    elem.classList.add('anno');
    elem.textContent = text_split[i];
    circle.append(elem);
    // ---

    let ANGLE = i * DELTA + 60;
    let RADIUS = R;
    let p = get_point(RADIUS, radians(ANGLE));
    circle.style.left = pxfy(p.x);
    circle.style.top = pxfy(p.y);
    // following transpositions applied to angles are entirely arbitrary.
    // use what you like or just use ${ANGLE}
    circle.style.backgroundColor = `hsl(${ANGLE * 2.3}deg, 90%, 70%)`;
    circle.style.color = `hsl(${ANGLE * 1.5 + 270}deg, 80%, 20%)`;

    // additional bells and whisles
    circle.style.transform = `rotate(${ANGLE + 90}deg)`;
```

```
    // ---

    c.appendChild(circle);
  });
};

setup();
```

```
.circle#main {
  --rad: 36px;
  position: absolute;
  width: 500px;
  height: 500px;
  top: 240px;
  left: 50%;
  transform: translate(-50%, -28%);
}
.circle#main div {
  position: absolute;
  border: 1px solid black;
  width: var(--rad);
  height: var(--rad);
  border-radius: var(--rad);

  /* everything below this is to style the text which can be removed */
  display: flex;
  justify-content: center;
  align-items: center;
  font-family: sans-serif;
  font-weight: bold;
  font-size: 11px;
  pointer-events: none;
  opacity: 1;
  transition: opacity 3s ease-in-out;
}

#main div > .anno {
  position: absolute;
  top: -2.8rem;
  font-size: 2rem;
  font-family: serif;
  text-align: center;
  text-transform: uppercase;
  font-weight: bold;
  text-rendering: geometricPrecision;
  border-color: white;
  border-bottom-color: currentColor;
}
```

Share Improve this answer Follow

edited Dec 1 at 16:44

answered Nov 28 at 17:48

lost-and-found

176 • 6