CSS+JS clock

Create a file containg the following code

clock.html

```
<article class="clock">
  <div class="hours-container">
    <div class="hours"></div>
  </div>
  <div class="minutes-container">
    <div class="minutes"></div>
  </div>
  <div class="seconds-container">
    <div class="seconds"></div>
  </div>
</article>
```

That’s right! Our clock consists only of divs.

Now Insert style into the file, otherwise we only have a page with lots of invisible divs.

```
.clock {
    position: fixed;
    left:50%;
    top:50%;
    margin-top:-150px;
    margin-left:-150px;
    height: 300px;
    width: 300px;
    border-radius: 50%;
    background: #fff url(test.svg) no-repeat center;
    background-size: 88%;
}

.clock.simple:after {
    background: #000;
    border-radius: 50%;
    content: "";
    position: absolute;
    left: 50%;
    top: 50%;
    transform: translate(-50%, -50%);
    width: 5%;
    height: 5%;
    z-index: 10;
}
```

```
.minutes-container, .hours-container, .seconds-container {
    position: absolute;
    top: 0;
    right: 0;
    bottom: 0;
    left: 0;
}

.hours {
    background: #000;
    height: 20%;
    left: 48.75%;
    position: absolute;
    top: 30%;
    transform-origin: 50% 100%;
    width: 2.5%;
}

.minutes {
    background: #000;
    height: 40%;
    left: 49%;
    position: absolute;
    top: 10%;
    transform-origin: 50% 100%;
    width: 2%;
}

.seconds {
    background: #000;
    height: 45%;
    left: 49.5%;
    position: absolute;
    top: 14%;
    transform-origin: 50% 80%;
    width: 1%;
    z-index: 8;
}

@keyframes rotate {
    100% {
        transform: rotateZ(360deg);
    }
}
.hours-container {
    animation: rotate 43200s infinite linear;
}
.minutes-container {
    transition: transform 0.3s cubic-bezier(.4,2.08,.55,.44);
}
.seconds-container {
```

```
    transition: transform 0.2s cubic-bezier(.4,2.08,.55,.44);
}
```

We could use animation for .minutes-container, and .seconds-container and we could have a smooth animation (and a correct one), but we want a little more personality for our clock. Simply adding steps to our animation makes the clock ‘tick’.

```
.minutes-container {
  animation: rotate 3600s infinite steps(60);
}
.seconds-container {
  animation: rotate 60s infinite steps(60);
}
```

Our clock hands are all pointing upwards and need to be angeled according to our local time.

```
/*
 * Starts clock using the user's local time
 */
function initLocalClocks() {
  // Get the local time using JS
  var date = new Date;
  var seconds = date.getSeconds();
  var minutes = date.getMinutes();
  var hours = date.getHours();

  // Create an object with each hand and it's angle in degrees
  var hands = [
    {
      hand: 'hours',
      angle: (hours * 30) + (minutes / 2)
    },
    {
      hand: 'minutes',
      angle: (minutes * 6)
    },
    {
      hand: 'seconds',
      angle: (seconds * 6)
    }
  ];
  // Loop through each of these hands to set their angle
  for (var j = 0; j &lt; hands.length; j++) {
    var elements = document.querySelectorAll('.' + hands[j].hand);
    for (var k = 0; k &lt; elements.length; k++) {
        elements[k].style.webkitTransform = 'rotateZ('+ hands[j].angle
+'deg)';
        elements[k].style.transform = 'rotateZ('+ hands[j].angle +'deg)';
        // If this is a minute hand, note the seconds position (to calculate
minute position later)
        if (hands[j].hand === 'minutes') {
```

```
          elements[k].parentNode.setAttribute('data-second-angle', hands[j +
1].angle);
        }
    }
  }
}
```

Since our clock isn't moving yet (except the hours handle wich is handeled by CSS animation), we need some JS to animate the minutes and seconds hands.

```
/*
 * Set a timeout for the first minute hand movement (less than 1 minute),
then rotate it every minute after that
 */
function setUpMinuteHands() {
  // Find out how far into the minute we are
  var containers = document.querySelectorAll('.minutes-container');
  var secondAngle = containers[0].getAttribute("data-second-angle");
  if (secondAngle > 0) {
    // Set a timeout until the end of the current minute, to move the hand
    var delay = (((360 - secondAngle) / 6) + 0.1) * 1000;
    setTimeout(function() {
      moveMinuteHands(containers);
    }, delay);
  }
}

/*
 * Do the first minute's rotation
 */
function moveMinuteHands(containers) {
  for (var i = 0; i < containers.length; i++) {
    containers[i].style.webkitTransform = 'rotateZ(6deg)';
    containers[i].style.transform = 'rotateZ(6deg)';
  }
  // Then continue with a 60 second interval
  setInterval(function() {
    for (var i = 0; i < containers.length; i++) {
      if (containers[i].angle === undefined) {
        containers[i].angle = 12;
      } else {
        containers[i].angle += 6;
      }
      containers[i].style.webkitTransform = 'rotateZ('+ containers[i].angle
+'deg)';
      containers[i].style.transform = 'rotateZ('+ containers[i].angle
+'deg)';
    }
  }, 60000);
}
```

```
/*
 * Move the second containers
 */
function moveSecondHands() {
  var containers = document.querySelectorAll('.seconds-container');
  setInterval(function() {
    for (var i = 0; i < containers.length; i++) {
      if (containers[i].angle === undefined) {
        containers[i].angle = 6;
      } else {
        containers[i].angle += 6;
      }
      containers[i].style.webkitTransform = 'rotateZ('+ containers[i].angle
+'deg)';
      containers[i].style.transform = 'rotateZ('+ containers[i].angle
+'deg)';
    }
  }, 1000);
}
```

And finaly do not forget to call the functions in the correct order:

```
initLocalClocks();
setUpMinuteHands();
moveSecondHands();
```