

# Seconds to midnight

When I read the assignment requirements the first thing that came to mind was the doomsday clock that I have heard about online and in the news in the past couple of months. The doomsday clock, which is represented in "seconds to midnight" represents how close humanity is to its destruction. This concept seems interesting to me as it feels both scary and intriguing when we find out we are only a minute or so away from the end of the world.

The idea I had in mind to bring this concept closer to all of the users who might've not heard about it before was to put the users in the shoes of the world leaders. You, as the visitor of the website, are able to make decisions such as going green, or regulating AI, or making nuclear negotiations with other leaders. But what you will realize is that all of these decisions have a cooldown, this is meant to represent the real world where these decisions aren't made overnight, rather there are many variables going into it. But to simplify the variables I decided to put a cooldown on the decisions, making the user slowly realize that they cannot beat the timer.

This project intends to show the urgency of actions we need to make in order to avoid reaching midnight. I try to make this symbolism more meaningful with some signs that the clock is getting closer to 0, such as the blinking that starts at 10 seconds left. Also the sudden change in the background at the end is meant to surprise the user, thus making the ending more meaningful and memorable.

## The process

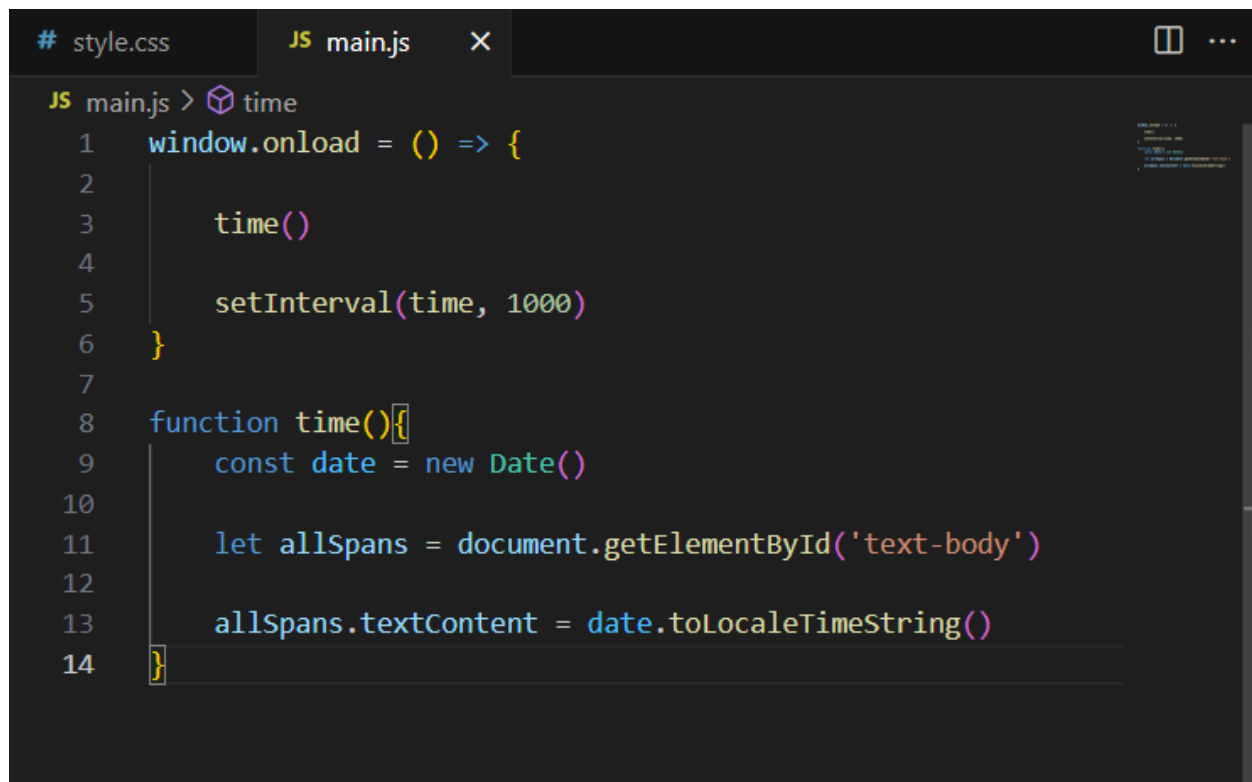
Before even starting to work on the website I had to research about the doomsday clock. How often does it change? What are the variables that "control" the clock. What can be done to stop the clock. All of this information can be found on the official website of the [Bulleting of Atomic Scientists](#). I would encourage everyone to go through the website themselves and learn more about the clock.

After research came the programming time!

```
index.html ×
1 <html lang="en">
2 <head>
3   <title>90 Seconds</title>
4   <link rel="stylesheet" href="style.css">
5   <script src="main.js"></script>
6 </head>
7 <body>
8   <h1>Seconds to midnight</h1>
9
10  <span id="text-body">hello!</span>
11 </body>
12 </html>

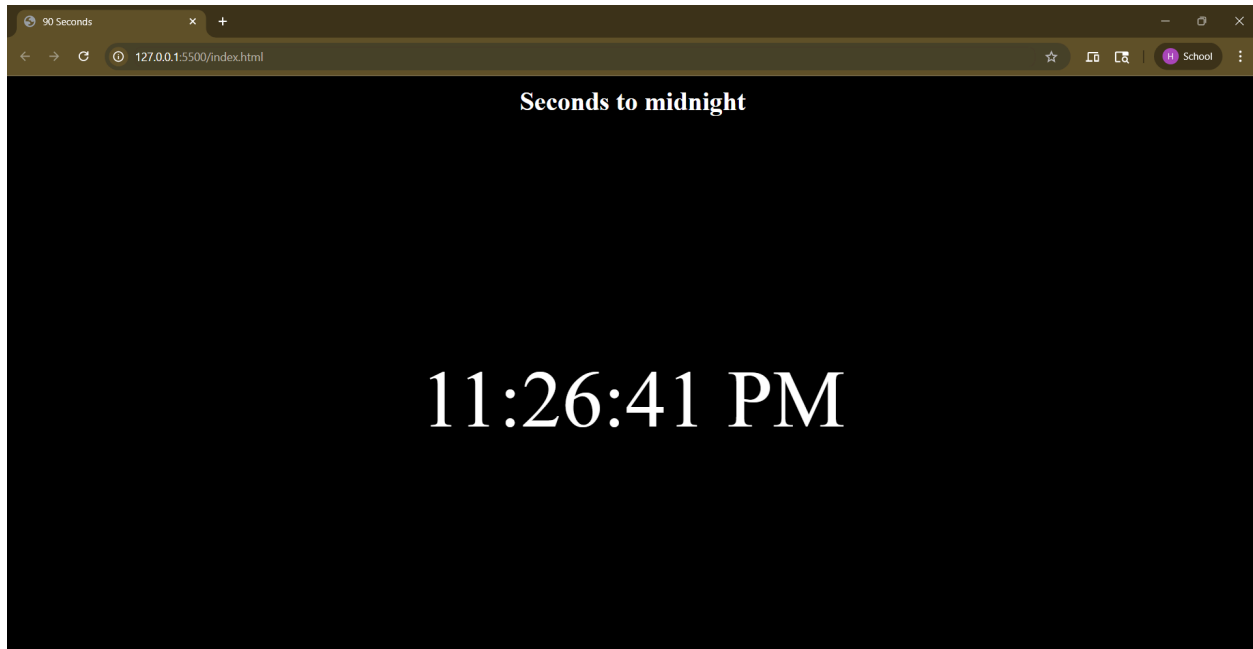
# style.css × JS main.js
# style.css > body
1 html{
2   background-color: black;
3   overflow: hidden;
4 }
5
6 body{
7   display: flex;
8   justify-content: center;
9   align-items: center;
10  flex-direction: column;
11  height: 100vh;
12 }
13
14 h1{
15   color: white;
16   text-align: center;
17   margin-top: 20px;
18 }
19
20 #text-body{
21   color: white;
22   margin: auto;
23   display: flex;
24   justify-content: center;
25   align-items: center;
26   font-size: 100px;
27 }
```

I started by first making a basic html structure with a placeholder text for the timer that is going to be placed on the screen.



```
# style.css JS main.js X
JS main.js > time
1 window.onload = () => {
2
3     time()
4
5     setInterval(time, 1000)
6 }
7
8 function time(){
9     const date = new Date()
10
11     let allSpans = document.getElementById('text-body')
12
13     allSpans.textContent = date.toLocaleTimeString()
14 }
```

After that it was time to work on the JavaScript side of the website. Initially I recreated the code we made in class with slight changes so that I can practice a little and get to know what I will be working with better. This is what the website looked at this time:



It was finally time to add the images. I used to be scared of working with flex containers as the concept seemed complicated to me. But after going through them deeper in class and with some practice I've realized that they are not that bad and are quite fun to work with and are incredibly useful!

```
index.html x
index.html > html
1 <html lang="en">
2 <head>
3   <title>90 Seconds</title>
4   <link rel="stylesheet" href="style.css">
5   <script src="main.js"></script>
6 </head>
7 <body>
8   <h1>Seconds to midnight</h1>
9
10  <span id="text-body">hello!</span>
11
12  <div id="imgs">
13    
14    
15    
16  </div>
17
18 </body>
19 </html>

# style.css x JS main.js
# style.css > #imgs
6 body{
7   justify-content: center;
8   align-items: center;
9   flex-direction: column;
10  height: 100vh;
11 }
12
13
14 h1{
15   color: white;
16   text-align: center;
17   margin-top: 20px;
18 }
19
20 #text-body{
21   color: white;
22   margin: auto;
23   display: flex;
24   justify-content: center;
25   align-items: center;
26   font-size: 60px;
27 }
28
29 #imgs{
30   display: flex;
31   justify-content: space-between;
32   align-items: center;
33   width: 100%;
34   padding: 0 40px;
35 }
36
37
38 img{
39   width: 300px;
40   height: 300px;
41 }
```

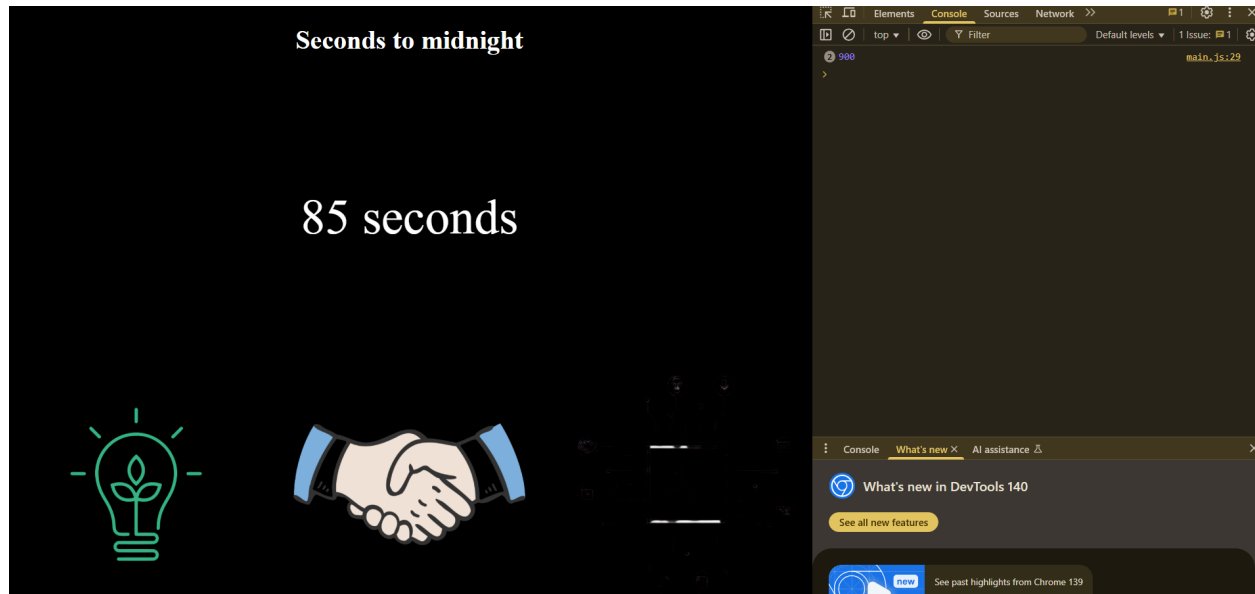
After getting everything set up and being happy with the looks, it was time to play with the functions and the JavaScript. Initially it was going very smoothly and I didn't encounter any issues that I couldn't fix just by reading the JS documentation. But after some time I realized I had one major issue. My initial idea for the website was that the timer would speed up every couple of seconds making it obvious that it is impossible to beat it and that no matter what the user does the outcome will be the same (that seems to be a pattern with my projects :)).

So I did what seemed logical to me, I made a variable that would hold the delay that the setInterval function takes and I would just make a function that would change that value after a certain time, or a certain number of clicks on the images. After doing so I was surprised to see that nothing was happening, everything seemed to be the same, even though the number should obviously be changing and my logic makes sense, right? RIGHT??

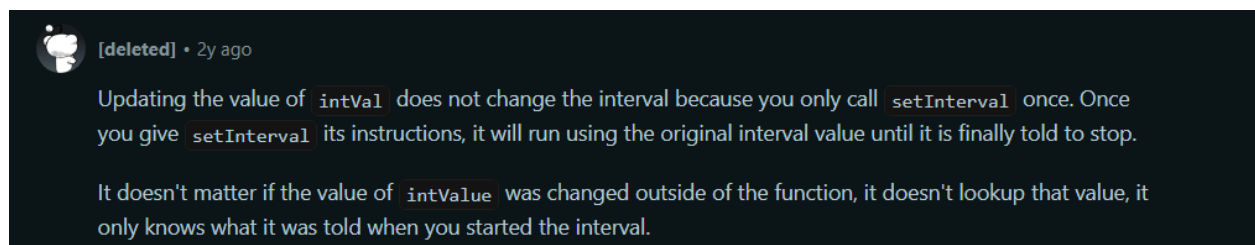
This made me very angry and I spent an hour playing with values and double, even triple checking the values I entered to see where I made a mistake. I only later realized that there was an extremely easy way to see what was going on and what would be a possible fix.

INSPECT ELEMENT

I feel ashamed for forgetting to do that and will surely never do so again after spending more than an hour debugging.



Now this is interesting. Even though the time has passed, and the images have been pressed multiple times, the console seems to be outputting the same number over and over again. Hmm... why could that be. After researching online I found out that the `setInterval` function is only called once so it will only use the value it was given initially.



This alongside other articles on the matter pointed out that the only way to have a dynamic delay was to use the `setTimeout` function. But since we were only allowed to use the `setInterval` function I had to ditch this idea ;(.

## References

The idea for the website came from the official doomsday clock website that was already listed above.

<https://thebulletin.org/doomsday-clock/>

For the time passing logic I was inspired by a game I played some time ago which is called "60 seconds!" I like how this game has you initially compete with a 60 seconds timer to collect resources before an atomic bomb eventually hits and you are forced to hide in your shelter. I remember being anxious looking at the timer and wanted to recreate something similar in my website.

<https://robotgentleman.com/60seconds/>

I was also inspired by a website game that was covered in one of my classes during my sophomore year. The game is basically about managing McDonalds resources with meat, and deforesting to make more room for parking and such. I liked how the game worked by clicking on images or objects to make moves and you also had a time pressure which really went well with my idea.

<https://molleindustria.org/mcdonalds/>

I also used various websites for documentation as I haven't used JavaScript in a while and needed some refreshing:

[https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Operators/Logical\\_AND](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Operators/Logical_AND)

[https://developer.mozilla.org/en-US/docs/Web/API/Element/click\\_event](https://developer.mozilla.org/en-US/docs/Web/API/Element/click_event)

[https://www.w3schools.com/js/js\\_if\\_else.asp](https://www.w3schools.com/js/js_if_else.asp)

<https://css-tricks.com/snippets/css/a-guide-to-flexbox/>

The images used were not created by me. Here are the links for them:

<https://cdn-icons-png.flaticon.com/512/10017/10017692.png>

<https://media.istockphoto.com/id/1300839614/vector/green-lightbulb-with-leaf-inside-clean-energy-concept.jpg?s=612x612&w=0&k=20&c=js1r3O1L-HZr5YZ0EZWPPIJIP8JJ1JmyDncrcDVqo=>

[https://static.vecteezy.com/system/resources/thumbnails/043/347/493/small\\_2x/concept-artificial-intelligence-ai-pictogram-technology-related-to-artificial-intelligence-idea-](https://static.vecteezy.com/system/resources/thumbnails/043/347/493/small_2x/concept-artificial-intelligence-ai-pictogram-technology-related-to-artificial-intelligence-idea-)

content-generator-illustration-free-vector.jpg

## Questions

I feel conflicted about the design. On one hand I chose simplistic design because I thought that the message was conveyed better that way, on the other hand it might be too simple? Would you recommend making it more complex and in what ways?

The users might initially be confused as to what is happening on the screen. I refrained from adding an explanation page due to assignment requirements. Do you think the explanation should be added with even one page? Maybe on one of the sides of the timer?

Do you think the right message is conveyed? Is the interaction meaningful at all?