Assignment 3: Animations

LIB2024: Internet Applications

Martha Kohler
040864508
March 20, 2023

Contents

Introduction

The purpose of this assignment was to create a web page which displayed at least one animation. I used CSS properties to achieve this goal and created a simple page featuring 2 animation elements.

Link to GitHub:

https://github.com/kohl0028/kohl0028.github.io/tree/main/lib2024/Assignment%203%20Martha%20Kohler

Elements and Tricks

First Animation: Color-Changing Text

To create a title/header for the page, I chose to animate the text with alternating colors. This was done using a template found on Codepen (Álvaro). I changed up the font and removed the extra coding for a fourth word, as it was not needed for my chosen title. A class selector "header" was defined for the and inside this, a tag was applied to each word. Since it is an inline element, it would serve to divide the text effectively without forcing a new line:

```
<div class="header">
  <h1 class="title">
    <span class="title-word title-word-1">Happy</span>
    <span class="title-word title-word-2">Holidays</span>
    <span class="title-word title-word-3">!</span>
  </h1>
</div>
```

For the CSS styling, the header container class is given a block display and the contents set to align to center, as well as margins set at automatic to centre the content horizontally (below, left). The title overall was given a font imported from Google and sized. The font link is included in of the HTML <head> (below, right).

```
.header {
  display: block;
  place-items: center;
  text-align: center;
  max-height: 25vh;
  margin: auto;
}
```

```
h1.title {
  font-family: 'Satisfy', cursive;
  font-weight: 800;
  font-size: 8.5vw;
}
```

Within the <h1>, the title-word class was given animation properties to define the type of animation, duration, and how many times to repeat (iteration count) – I wanted the text to continue changing so I chose infinite (see below, left). Next, each word was styled with 3 colors to cycle through, for example word one as shown below. Words 2 and three were given the same treatment (below, right).

```
.title-word {
  animation-name: color-animation;
  animation-duration: 4s;
  animation-iteration-count: infinite;
}
```

```
.title-word-1 {
  --color-1: #54A88A;
  --color-2: #AACE71;
  --color-3: #E4A9A8;
}
```

```
@keyframes color-animation {
  0%    {color: var(--color-1)}
  32%   {color: var(--color-1)}
  33%   {color: var(--color-2)}
  65%   {color: var(--color-2)}
  66%   {color: var(--color-3)}
  99%   {color: var(--color-3)}
  100%  {color: var(--color-1)}
}
```

Finally, the @keyframes property is used to define at what point the colors switch. Along certain percentages, the next color is applied, with 0% and 100% given the same color to create a seamless transition when the animation cycle is repeated.

Second Animation: Snow Globe

The snow globe animation was inspired by a script found on Codepen (Coding Artist). I thought the drifting snow was neat and wanted to use it for my festive page. The animation was interesting since the entire image is 'drawn' using CSS style properties to create each shape. Instead of just copying the existing code, and to pare down the style sheet, I decided to substitute an image of my own choosing for the house. I still used the script for the shape of the snow globe and the animation.

The HTML coding for this element was quite straightforward; I wrapped all the graphic elements into a container :

```
<div class="container">
  <div class="snowfall"></div>
  <div class="cover"></div>
  <div class="bottom">
    <div class="bt1"></div>
    <div class="bt2"></div>
    <div class="house"> <img src="house.png" alt="House" width="175px"></div>
  </div>
</div>
```

The CSS gets more complex as each child element needs to be individually rendered. First, the parent container was styled. Margins and padding were applied, as well as a height and width for the image (since all the child elements are positioned relative to the parent container, these measurements needed to be defined. Finally, border-radius was used to round the corners. In this case, since the height and width are the same, the border becomes a circle when 50% is applied.

```
.container {
  margin: 0 auto;
  padding: 500px;
  height: 350px;
  max-width: 350px;
  border-radius: 50%;
}
```

Next, the snowfall class styled the snowflakes, which was a linked background image. The background size was defined and repeated to fill the entire space, and then the border radius was again set at 50%

to create the rounded shape. The animation shorthand defined the style, timing, movement, and direction and set to an infinite loop.

```
.snowfall{
  background: url(https://i.imgur.com/T0mBFg2.png);
  background-size: 350px 200px;
  background-repeat: repeat;
  height: 350px;
  width: 350px;
  position: relative;
  bottom: 493px;
  border-radius: 50%;
  animation: snowfall 5s linear forwards infinite;
}
```

The @keyframes property was applied to further describe the change and how the animation will appear. Progress points every 20% from 0 to 100 were identified and set the image to move down and left 20px each, ultimately displaying a diagonal movement of the image. Set on a loop, the background static image of snowflakes appears to drift peacefully down inside the globe. (See image, right)

```
@keyframes snowfall{
  0%{
    background-position: 0 0;
  }
  20%{
    background-position: 20px 20px;
  }
  40%{
    background-position: 40px 40px;
  }
  60%{
    background-position: 60px 60px;
  }
  80%{
    background-position: 80px 80px;
  }
  100%{
    background-position: 100px 100px;
  }
}
```

The other, static elements of the snow globe (cover, bottom, bt1 and bt2) were rendered using CSS styling, and the house image was inserted into the div and styled.

All the child elements were positioned relative to the parent container, so they stack appropriately to create the overall snow globe illustration (see images below):

```
.cover{
  background-color: rgba(255,255,255,0.2);
  height: 350px;
  width: 350px;
  border-radius: 50%;
  position: relative;
  bottom: 842px;
}
.bottom{
  background-color: #c1272e;
  height: 70px;
  width: 260px;
  position: relative;
  bottom: 880px;
  left: 47px;
}
```

```
.bt1,.bt2{
  background-color: #d33c3c;
  height: 20px;
  width: 280px;
  position: relative;
  right: 10px;
  border-radius: 10px;
}
.bt1{
  bottom: 12px;
}
.bt2{
  top: 40px;
}
```

```
}
.house img{
  position: relative;
  bottom: 290px;
  left: 44px;
  z-index: -1;
}
```

In addition to defining the positioning, the house image was given a z-index of -1 so it would sit at the bottom of the layered images to give a more three-dimensional appearance to the animation (so the snowflakes would drift in front).

Troubleshooting

In positioning the snow globe container as a whole, I had trouble and cold not figure out how to keep it centered if the window size narrows. The snow globe hugs the right side and the title overlaps the image as the page is condensed. This is likely due to the relative positioning of the stacked CSS properties created to render the image within the . Perhaps applying a grid layout to the entire page would solve this issue, and keep all elements appropriately positioned. It is difficult when stitching in bits of code from various sources, for them to act as anticipated once plugged into a larger page element. For the purposed of this assignment, which was to effectively render animations, I didn't worry about this as much and left the element as is. I was afraid to mess up how the child elements interacted with each other.

Accessibility & SEO

For this project, the header is the main place where SEO is met. The metadata includes important information for search engines to scan - such as page description, content, keywords, author, etc. Choosing appropriate wo

In terms UX and accessibility, in the header the viewport is established to adjust to device width, so multiple screen sizes should be able to read this page effectively. The image elements were given alternate descriptions which would display should the linked graphic fail to load. Since this is a simple page without a lot of content, more of a landing page than informative website, I was not able to apply most of the standards such as semantic HTML or structural elements to organize content.

I chose colours which are not too loud or bright, especially since they will flash on a constant loop. The rate at which the colors change, and thee text colors and large size provide sufficient contrast (as per WCAG AA standards, shown on WebAIM's Contrast Checker).

Finally, the page has a simple layout with plenty of empty space to rest the eye.

Conclusion

The goal of the assignment was to build a page with animations using at least HTML and CSS, which has been accomplished. Layering and positioning elements continues to be challenging and I think I would benefit from planning layouts from the start to be a better approach than simply trying to plug in snippets of code and then trying to position them.

References

Álvaro [username]. *Change the colors of each word – CSS Animation*. Codepen. https://codepen.io/alvarotrigo/pen/jOLgeqe

Coding Artist [username]. *Christmas Snow Globe Animation.* Codepen. https://codepen.io/Coding-Artist/pen/xmdOxB

Lebedeva, Juliya, *House Clipart Christmas - Christmas House Free Clipart, HD Png Download*. PNGItem. https://www.pngitem.com/middle/immwJwT_house-clipart-christmas-christmas-house-free-clipart-hd/

Sideshow [username]. *Satisfy.* Google Fonts. https://fonts.google.com/specimen/Satisfy

W3 Schools. *CSS @keyframes Rule*. https://www.w3schools.com/cssref/css3_pr_animation-keyframes.php