ISCG6420 Semester 1 2024

Exercise: Turn your storyboard into a CSS Animation

This exercise will walk you through creating most of a website advertisement using CSS.
It can be implemented in a new webpage or integrated into an existing webpage.

Setup the webpage advertisement space.

Create a new container element in your webpage HTML. You can use any suitable semantic or generic element according to your desired webpage structure. The example provided uses a <div> generic element for this purpose.

```html
<section>
  <div id="adWrapper">

  </div>
</section>
```

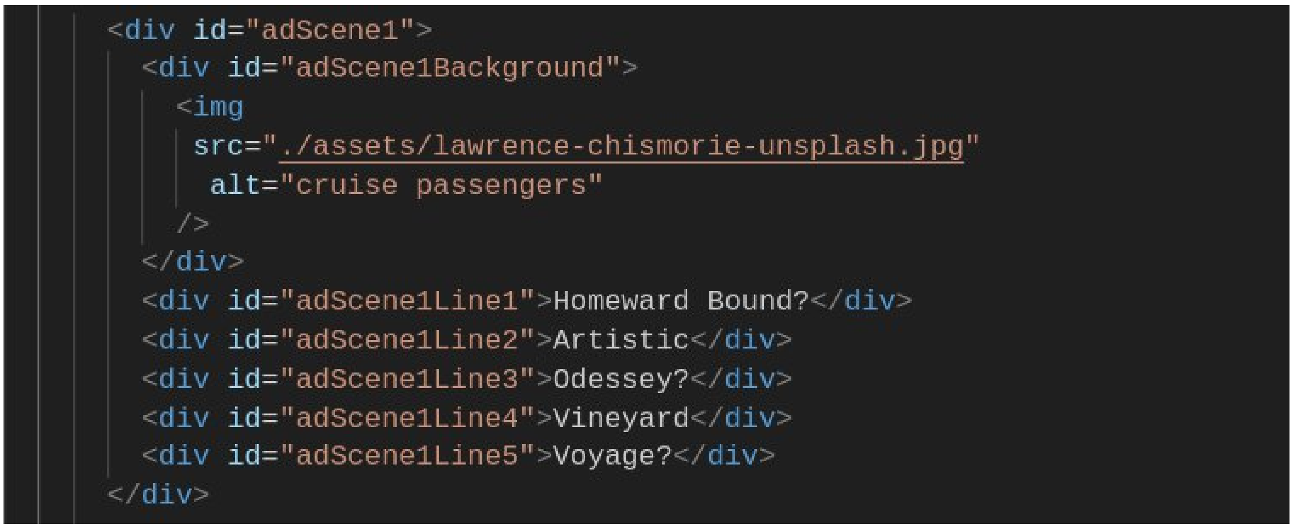
Create the first scene

We can create each scene of the advertisement as a separate container element to isolate the components of each scene. This will make organising and animating in CSS easier.
We can use a <div> with an ID.

```html
<section>
  <div id="adWrapper">
    <div id="adScene1">

    </div>

    <!-- other scenes go below -->
    <div id="adScene2">

    </div>
  </div>
</section>
```

Add the scene elements

Create a new HTML element for each component of the scene. Here we use <div> elements with IDs to provide access to each element individually. Note the use of meaningful ID names. While long variable names are not really recommended due to overutilisation of line space, readable and understandable code takes priority over name length.

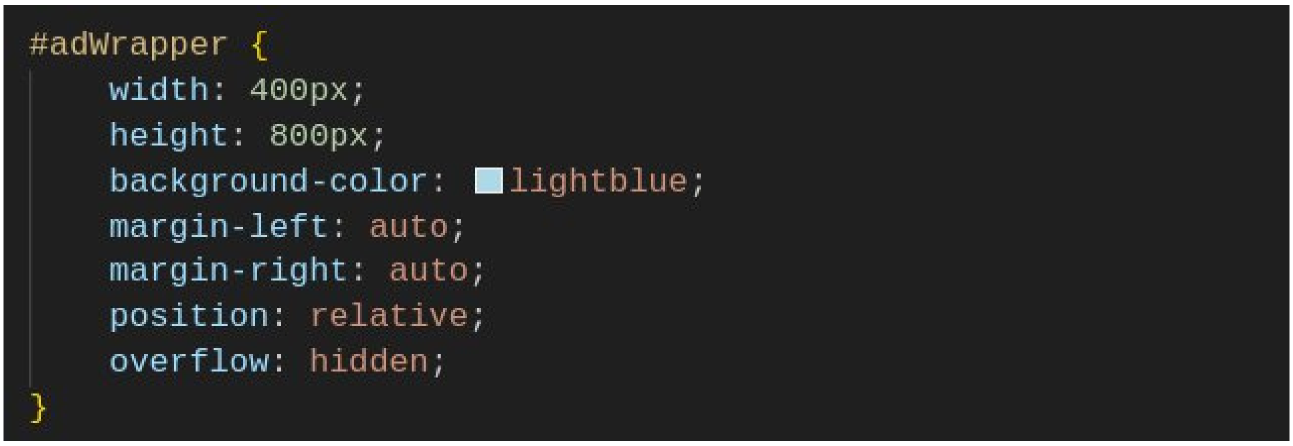


```
<div id="adScene1">
  <div id="adScene1Background">
    <img
     src="./assets/lawrence-chismorie-unsplash.jpg"
      alt="cruise passengers"
    />
  </div>
  <div id="adScene1Line1">Homeward Bound?</div>
  <div id="adScene1Line2">Artistic</div>
  <div id="adScene1Line3">Odessey?</div>
  <div id="adScene1Line4">Vineyard</div>
  <div id="adScene1Line5">Voyage?</div>
</div>
```

For the background of the scene we have used a <div> enclosed <img> with a free-use image from unsplash that roughly matches the theme we are replicating.

Style the wrapper

In your CSS file, add the following style to the advertisement container element:



```
#adWrapper {
    width: 400px;
    height: 800px;
    background-color: lightblue;
    margin-left: auto;
    margin-right: auto;
    position: relative;
    overflow: hidden;
}
```

Note: the margin-left and margin-right auto value will centre the element within its parent element. This can be applied to almost any element that you wish to centre. If you need to centre the contents of an element, you can use text-align: center instead.

IMPORTANT: Set the position to relative for any container where you wish to have overlapping or stacked child elements. For each child, set their position to absolute.

Style the background

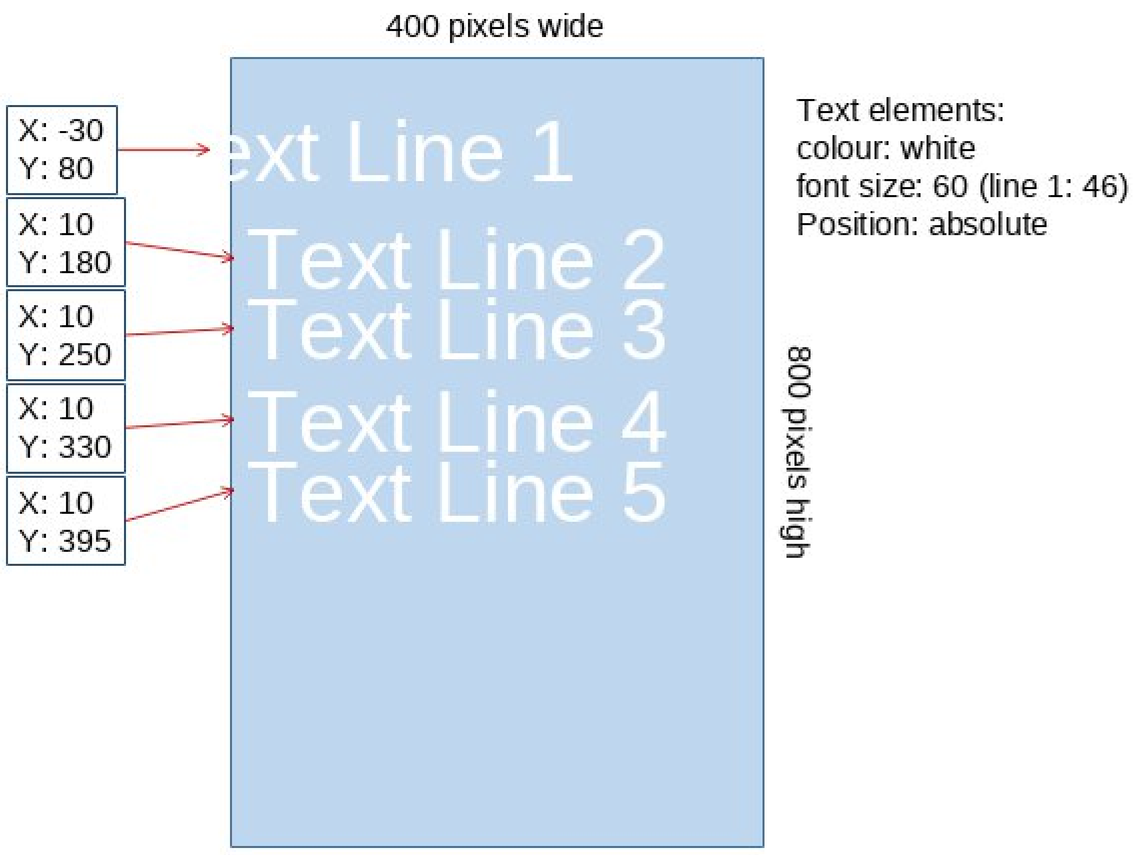
Apply the following style to the background image <div>:

```
#adScene1Background img {
    position: absolute;
    height: 1050px;
    left: -531px;
    top: -60px;
}
```

The size of the background image is larger than the ad-wrapper. That's okay, in fact we actually need to be larger for our animation effects later.

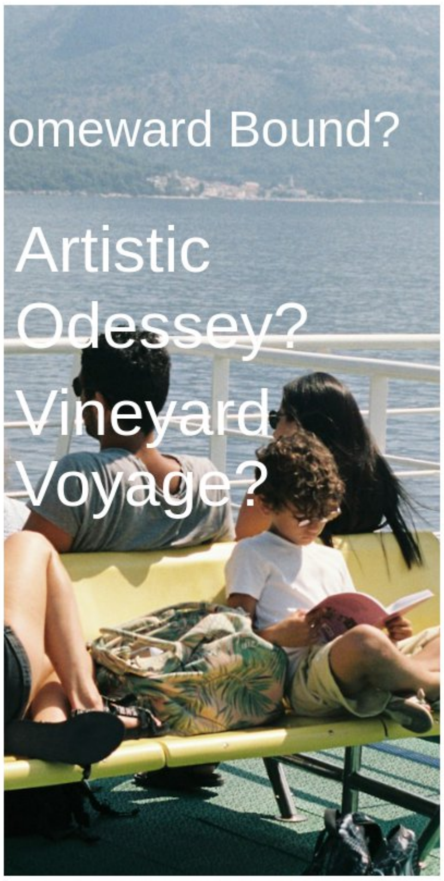
Style the text elements

Apply styling to the text line elements to comply with the following diagram:



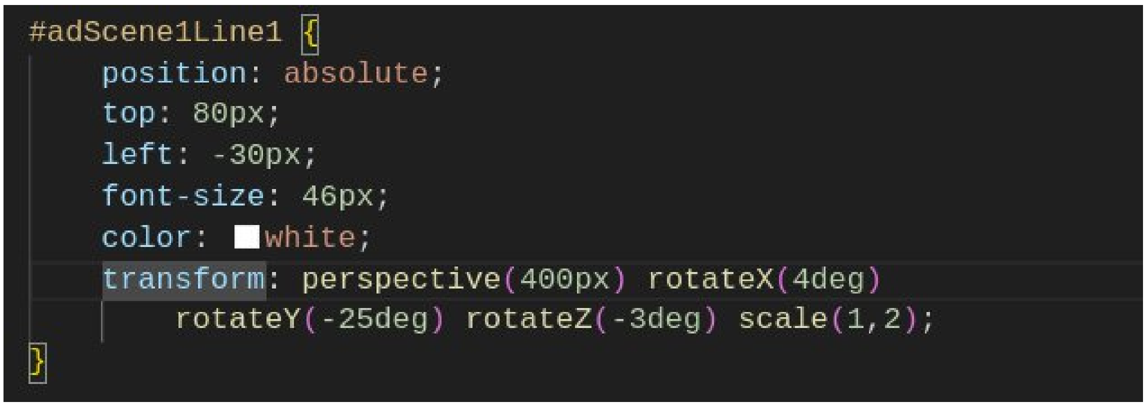
Remember to use the **top** and **left** attributes for X and Y positioning in CSS.
More positioning examples here: https://www.w3schools.com/Css/css_positioning.asp

Your advertisement should look like this so far:



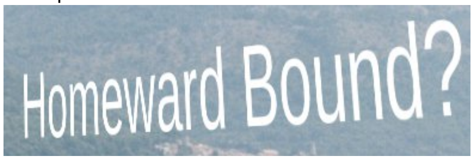
Apply text effects

To replicate (as best we can for now) the stylized text from the source material, apply the following transform to the line1 <div> style:



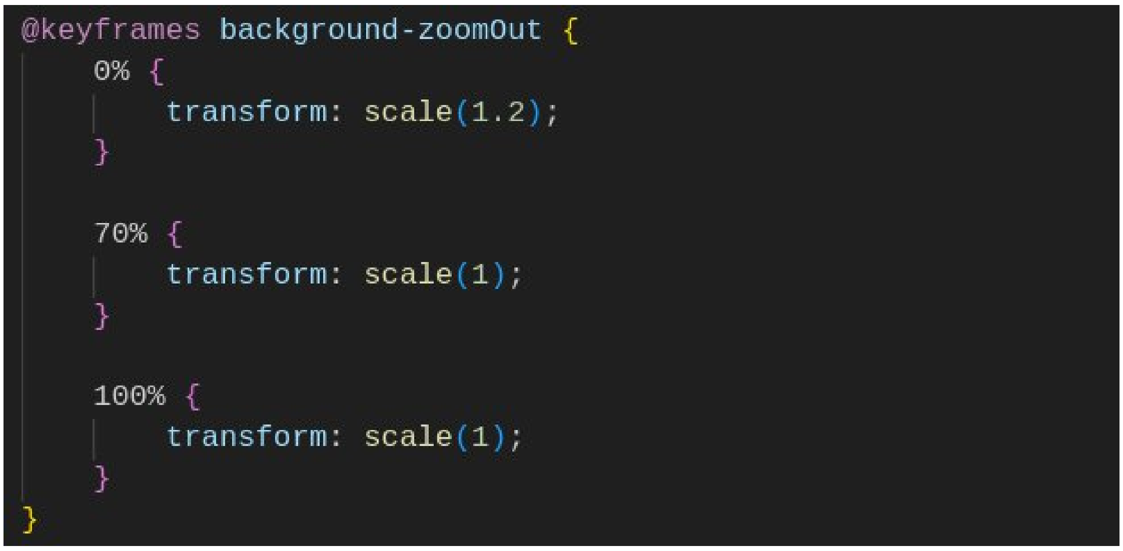
```
#adScene1Line1 {
    position: absolute;
    top: 80px;
    left: -30px;
    font-size: 46px;
    color: ■white;
    transform: perspective(400px) rotateX(4deg)
        rotateY(-25deg) rotateZ(-3deg) scale(1,2);
}
```

We are combining multiple transforms to create the final text effect. It's not perfect but it gives 80% of the desired result for 20% of the effort required to complete it. You can adjust the provided values to see how each effects the output.



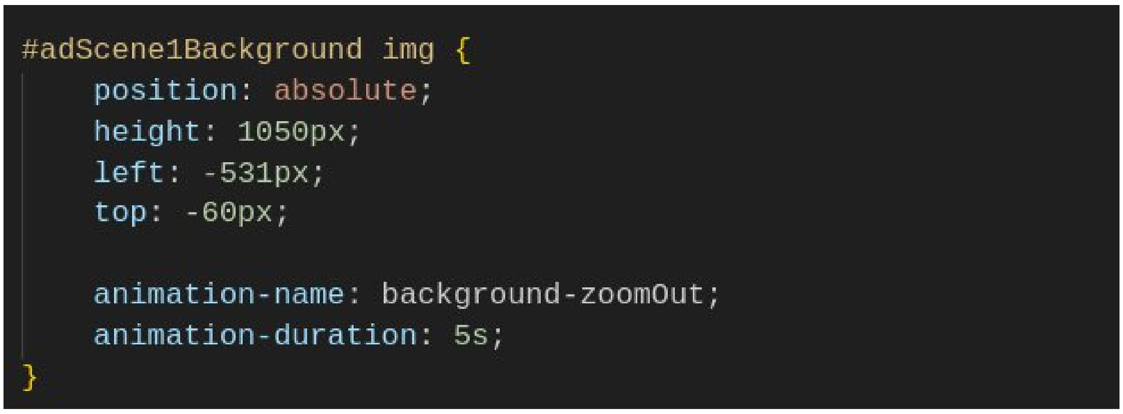
Recreate the scene zoom out animation

The slow zoom out effect can be achieved by changing the scale of the background image. Create a new animation keyframes block and give it a suitable name. Create three states: 0%, 70%, and 100%:



```css
@keyframes background-zoomOut {
    0% {
        transform: scale(1.2);
    }

    70% {
        transform: scale(1);
    }

    100% {
        transform: scale(1);
    }
}
```

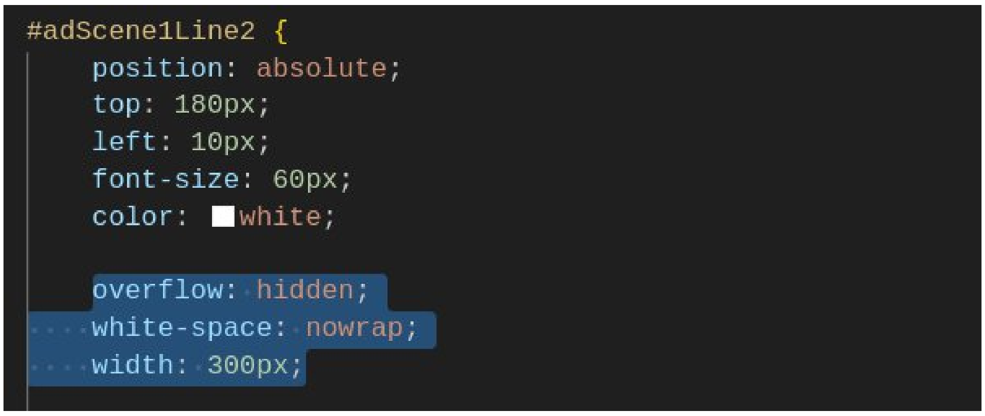
Apply the keyframes to the background element with the following styles changes:



```css
#adScene1Background img {
    position: absolute;
    height: 1050px;
    left: -531px;
    top: -60px;

    animation-name: background-zoomOut;
    animation-duration: 5s;
}
```

Recreate the text unveil animation

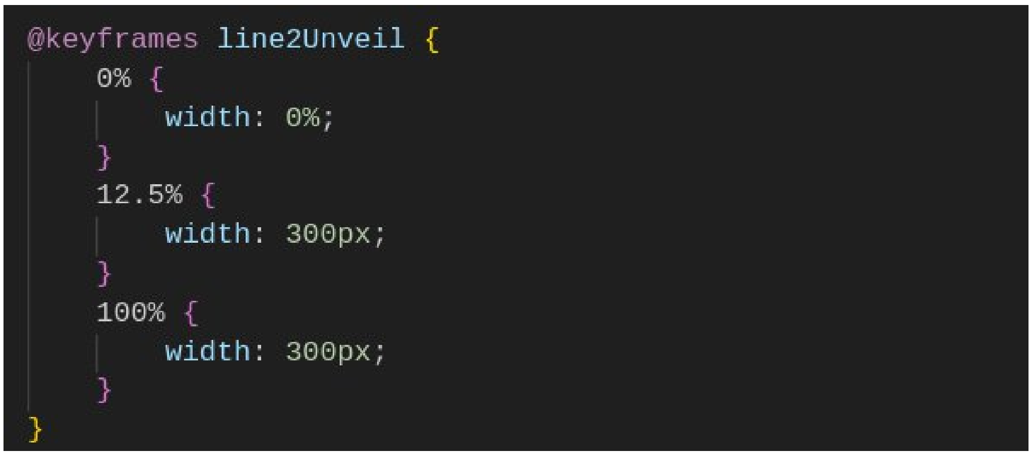
The unveiling of text, one line at a time, can be achieved by manipulating the way text behaves within its parent element, and the dimensions of the parent. Make the following changes to Line 2's styling:



```css
#adScene1Line2 {
    position: absolute;
    top: 180px;
    left: 10px;
    font-size: 60px;
    color: white;

    overflow: hidden;
    white-space: nowrap;
    width: 300px;
```

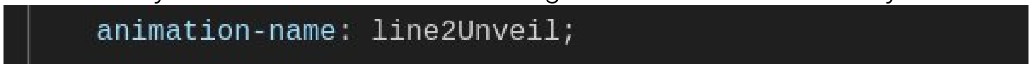
This will ensure the shape of the text doesn't change when the parent does. We can now change the line width with animation.

Create a new animation keyframes block and give it a suitable name. Create three states: 0%, 12.5%, and 100%:



```css
@keyframes line2Unveil {
    0% {
        width: 0%;
    }
    12.5% {
        width: 300px;
    }
    100% {
        width: 300px;
    }
}
```

Link the keyframes to the element using the animation-name style attribute:



```css
    animation-name: line2Unveil;
```

Repeat the changes made to line 2 (text behaviour and keyframes) to lines 3, 4, and 5.

Animate the scene

Apply the following styles to the scene 1 <div> container:

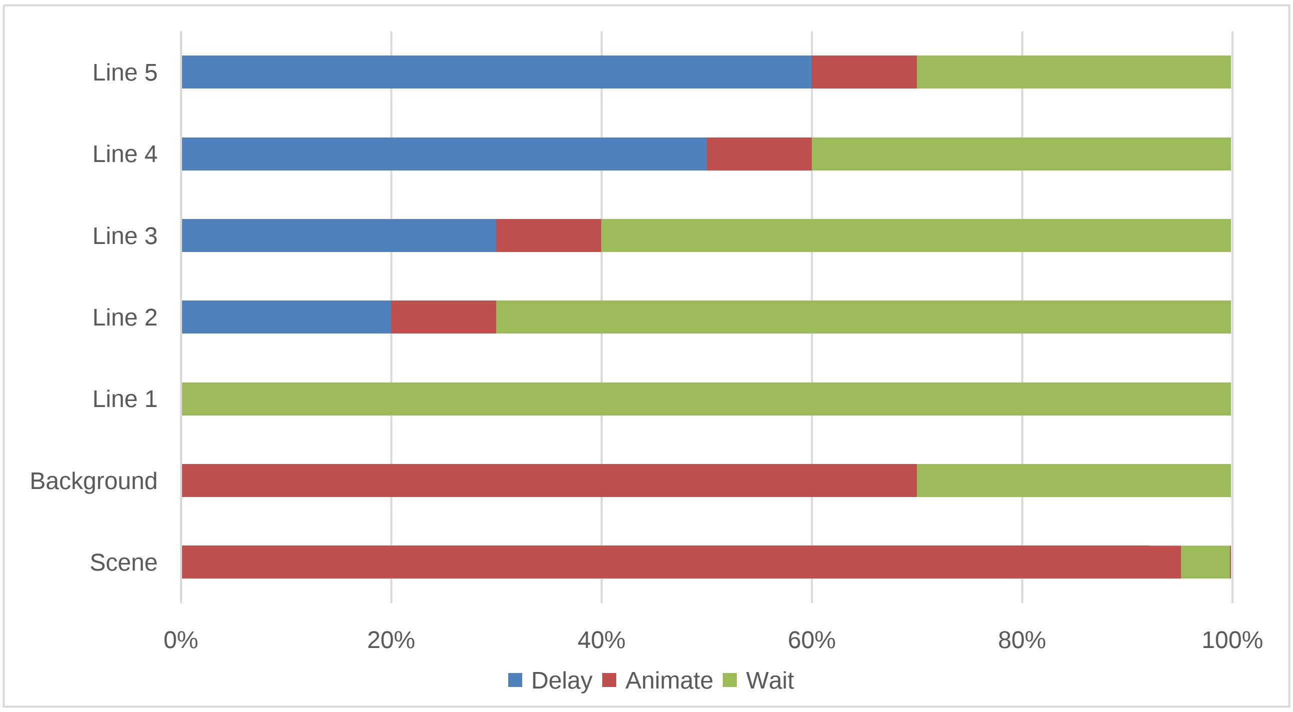
```css
#adScene1 {
    visibility: visible;
    animation-name: scene1;
    animation-duration: 5s;
}
```

```css
@keyframes scene1 {
    0% {
        visibility: visible;
    }

    95% {
        visibility: visible;
        opacity: 1;
    }

    100% {
        visibility: hidden;
        opacity: 0;
    }
}
```

We now need to orchestrate the scene elements to animate within the scene timeline.

Animaton orchestration

Scene 1 has a timeline similar to this:



Which means the elements should have the following animation time values:

	Delay	Animation	Wait	Animation midstep %
Scene	0	4.75	0	95
Background	0	3.5	1.5	70
Line 1	0	0	0	N/A
Line 2	1	0.5	3.5	12.5
Line 3	1.5	0.5	3	14.3
Line 4	2.5	0.5	2	20
Line 5	3	0.5	1.5	25

Set the delay and animation time of each element to their respective values like this:

*Line 2: delay for 1 second, animate for 0.5 seconds, then hold the animation for 3.5 seconds. Duration = animation + wait. Keyframe midstep% = 100 / duration * animation*

```
  animation-name: line2Unveil;
  animation-duration: 4s;
  animation-delay: 1s;
```

```
@keyframes line2Unveil {
    0% {
        width: 0%;
    }
    12.5% { /* this line is the mid-step */
        width: 300px;
    }
    100% {
        width: 300px;
    }
}
```

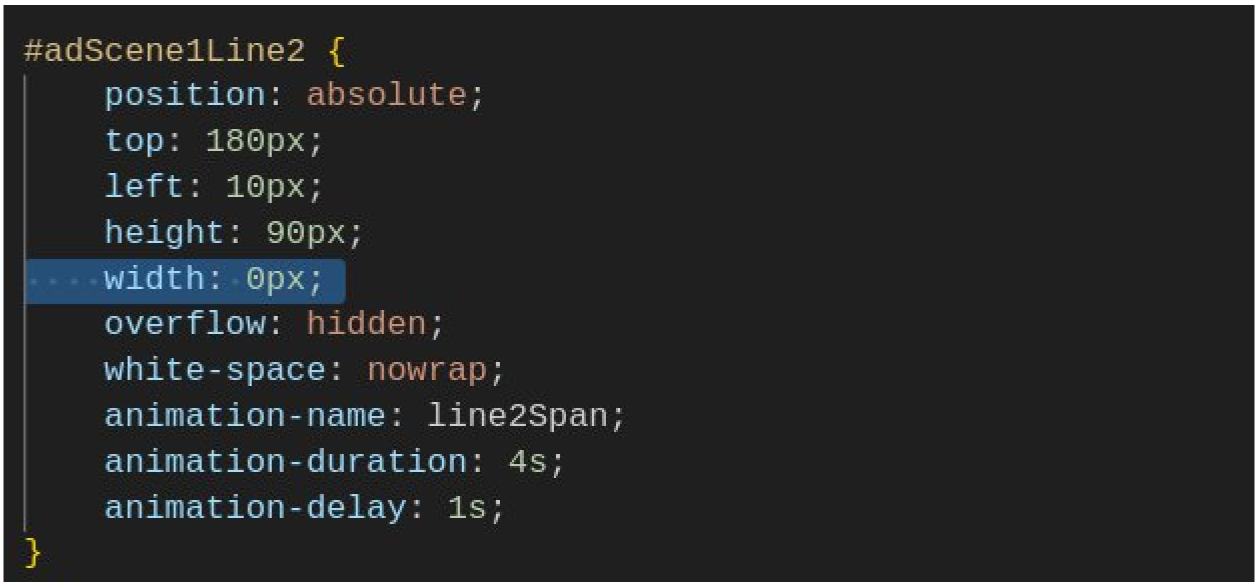
Fixing element pop-back

When each element finishes animating it will return to its pre-animation state. This means elements that are visible by default and animated to disappear will reappear after the animation completes. We can resolve this by changing the default state of such elements.

For the scene, set the default visibility to hidden:

```
#adScene1 {
    visibility: hidden;
    animation-name: scene1;
    animation-duration: 5s;
}
```

For the animated text elements, set the default width to 0px:



```css
#adScene1Line2 {
    position: absolute;
    top: 180px;
    left: 10px;
    height: 90px;
    width: 0px;
    overflow: hidden;
    white-space: nowrap;
    animation-name: line2Span;
    animation-duration: 4s;
    animation-delay: 1s;
}
```

Acknowledgements and further steps

Most of scene 1 is now complete. There are a few things that require some attention to complete the advertisement:

- The text font is not similar to the source advertisement (easy)
- Scene two is missing (moderate)
- The animated text elements are not transformed like the first text line (hard)

Use these as goals to accomplish with self-directed learning.

Exercise complete.