# Lesson 8

The focus of this lesson will be to help you further develop your portfolio website. Special consideration will be given to:

- Ability to present your portfolio item.
- Accessibility of your portfolio items.

Considerations also given to learning code by creating annotation.

## Video Recordings:

https://vimeo.com/715520536/5a42f2be61

https://vimeo.com/715546707/d47798fd7a

https://vimeo.com/715548660/5af8830e20

## Tasks:

- Create a content element that is hidden until it is targeted using its ID reference in the URL.
- Use this feature to for multiple elements that can be navigated to from <a> links.
- Animate this element using either **transition** or **animation**.

## Example 1:

This is the completed code from yesterday's content search feature. This version includes the ability to reset the search results so that previous results are not mixed with new searches.

```
<!DOCTYPE>
<html>
  <head>
  </head>
  <body>

    <select onChange="search(this.value)">
      <option value="">All</option>
      <option value="javascript">Javascript</option>
      <option value="html">HTML</option>
      <option value="css">CSS</option>
    </select>

    <ul>
      <li data-search="html css">Item 1</li>
      <li data-search="css">Item 2</li>
      <li data-search="javascript html">Item 3</li>
      <li data-search="javascript">Item 4</li>
    <ul>

<style>
.filtered > *:not(.active){
  display: none;
}
</style>

<script>
function search(value){
  alert(value);
  let parent = document.querySelector("ul")
  parent.classList.add("filtered");

  search.reset(parent);

  if(value != ""){
    document.querySelectorAll(`ul > [data-search~="${value}"]`).forEach(function(item){
      console.log(item);
      item.classList.add("active");
    });
  }else{
    parent.classList.remove("filtered");
  }

}

search.reset = function(container){
  container.querySelectorAll(".active").forEach(function(item){
    item.classList.remove("active");
  });
}
</script>

  </body>
</html>
```

## Example 2:

This example shows how content can be revealed using a CSS animation triggered by a content element's ID being referenced in the URL. It also shows how responsive images can be defined using CSS.

```
<!DOCTYPE>
<html>

  <head>
  </head>
  <body>

    <h1>My Portfolio</h1>
    <p>My introduction here...</p>

    <a href="#content1" class="image" style="background-image:url(https://www.w3schools.c
      data-reveal="content1">Show Content 1</a>
    <a href="#content2" class="image" style="background-image:url(https://www.w3schools.c
      data-reveal="content1">
      <span>Show Content 2</span>
    </a>

    <div id="content1" class="reveal">
      <h2>Hello!</h1>
      <p>This is content...</p>
    </div>

    <div id="content2" class="reveal">
      <h2>Bonjour!</h1>
      <p>This is more content...</p>
    </div>

<style>
.reveal{
  height: 0;
  opacity: 0;
  overflow: hidden;
  transition: height 1s, opacity 1s;
}

.reveal:target{
  height: 50vh;
  opacity: 1;
}

.image{
  display: inline-block;
  width: 15em;
  height: 15em;
  margin: .5em;
  background: no-repeat;
  background-size: cover;
  background-position: 50% 50%;
  border: 1px sold #000;
  color: #fff;
  text-decoration: none;
}
a.image:hover{
  opacity: .75;
}
.image > *{
  display: block;
  background: blue;
}
</style>

  </body>
</html>
```