Web developer setup and resources used in the section lessons.

jQuery CDN https://developers.google.com/speed/libraries
Placeholder images https://placeholder.com/
Editor code https://code.visualstudio.com/

Developer Tools and Page Setup Create Index Page

Section setup details and info for starting the project - resource and websites.

Lesson Challenge

1. Create index and js files
2. Setup basic HTML structure
3. Link to jQuery Library check to ensure its ready
4. Open editor write some HTML

```
<!DOCTYPE html>
<html>
 <head><title>jQuery Slider Project</title></head>
 <body>
   <h1>jQuery Slider Project</h1>
   <div class="container">
     <img src="https://via.placeholder.com/250/0000FF/808080
?Text=Hello">
   </div>
   <script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.m
in.js"></script>
   <script src="app.js"></script>
 </body>
</html>
```

Create an array of Image Paths used to output images to the Page

Explore how to setup a testing array of placeholder images, customize the images and generate a useful array of content for your page.

Lesson Challenge

1. Generate an array of image paths
2. Add images to the page
3. Add active class to the first image
4. Wrap img tag with a div and add a span for captions
5. Load onto the page with document ready

```
const arr = [];
const props = {
   "main" : $(".container")
};

$(document).ready(function(){
   console.log('doc ready');
   makelistImages();
   outputImages();
})

function outputImages(){
   $.each(arr,function(index,value){
       console.log(value);
       let tempActive = index == 0 ? 'active': '';
       console.log(tempActive);
```

```
        let html = `<div class="slide ${tempActive}"><img
src='${value}'><span>Caption ${index+1}</span></div>`;
        props.main.append(html);
    })
}


function  makelistImages(){
    for(let x=0;x<5;x++){
        let temp =
`https://via.placeholder.com/400x400/33ee33/22ff22?text=image${x
}`;
        arr.push(temp);
    }
}

/*
$(window).on("load",function(){
    console.log('window ready');
})

$(window).load(function(){
    console.log('window ready');
}) */
```

Random Image Colors function adding colorful images to the web page.

Using JavaScript String methods create random hex color value generator.

Lesson Challenge

1. Using String methods generate random hex values that can be used for color values
2. Convert random number to hex string value with toString(16)
3. Get 6 character hex value that can be used as a color value with string method for substr(2,6)
4. Update the random image paths with color values.

```
const arr = [];
const props = {
   "main": $(".container")
};

$(document).ready(function () {
   console.log('doc ready');
   makelistImages();
   outputImages();
})

function outputImages() {
   $.each(arr, function (index, value) {
       //console.log(value);
       let tempActive = index == 0 ? 'active' : '';
       //console.log(tempActive);
       let html = `<div class="slide ${tempActive}"><img
src='${value}'><span>Caption ${index+1}</span></div>`;
       props.main.append(html);
   })
```

```
}



function makelistImages() {
   for (let x = 0; x < 5; x++) {
       let temp =
`https://via.placeholder.com/400x400/${rClr()}/${rClr()}?text=Im
age ${x}`;
       arr.push(temp);
   }
}



function rClr(){
   const temp = Math.random();
   console.log(temp);
   const temp1 = temp.toString(16);
   console.log(temp1);
   const temp2 = temp1.substr(2,6);
   console.log(temp2);
   return temp2;
   // return Math.random().toString(16).substr(2,6);
}
```

Add CSS Styling to jQuery Image Slider Project

Make it look nice - setup the images to stack and position images to prepare for jQuery Coding. Create visible images output to page and setup CSS.

Lesson Challenge

1. Apply Styling to the project
2. Stack images within one main element
3. Use zindex to show first image only
4. Use opacity to hide the other images that are not active
5. Set widths for the image slider, also add the caption area with styling for anticipated content.

```
<!DOCTYPE html>
<html>
 <head><title>jQuery Slider Project</title>
 <link rel="stylesheet" href="style.css">
 </head>
 <body>
   <h1>jQuery Slider Project</h1>
   <div class="container">
    <div class="slideArea"></div>
   </div>
   <script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.m
in.js"></script>
   <script src="app.js"></script>
 </body>
</html>


const arr = [];
const props = {
   "main": $(".slideArea")
};

$(document).ready(function () {
```

```
    console.log('doc ready');
    makelistImages();
    outputImages();
})

function outputImages() {
    $.each(arr, function (index, value) {
        //console.log(value);
        let tempActive = index == 0 ? 'active' : '';
        //console.log(tempActive);
        let html = `<div class="slide ${tempActive}"><img
src='${value}'><span>Caption ${index+1}</span></div>`;
        props.main.append(html);
    })
}

function makelistImages() {
    for (let x = 0; x < 8; x++) {
        let temp =
`//via.placeholder.com/400x400/${rClr()}/${rClr()}?text=Image
${x}`;
        arr.push(temp);
    }
}

function rClr(){
    const temp = Math.random();
    //console.log(temp);
```

```
    const temp1 = temp.toString(16);
    //console.log(temp1);
    const temp2 = temp1.substr(2,6);
    //console.log(temp2);
    return temp2;
    // return Math.random().toString(16).substr(2,6);
}



*{
    box-sizing: border-box;
}

.container{
    width:680px;
    margin:0 auto;
}
.slideArea {
    margin: 0 10px;
    position: relative;
    width:680px;
    height:500px;
    background-color: black;
}

.slide {
    position: absolute;
    opacity: 0;
```

```
    z-index: 10;
    padding:10px;
    text-align: center;
}

.slide img{

    width: 660px;
    height:480px;
    overflow: hidden;
}

.slide span{
    background-color: rgba(0,0,0,0.5);
    color:white;
    display: block;
    position: absolute;
    bottom:30px;
    width:100%;
    left:0px;
    padding:15px;
    text-transform: uppercase;
    font-size: 1.5em;
    overflow: hidden;

}

.active{
```

```
    opacity: 1;
    transition: opacity 1s ease;
    z-index: 11;
}
```

Adding and removing element classes with jQuery. Setting Interval moving slides.

Using selection of active elements - to update and remove classes adding new active class to the new sibling element using jQuery Code. Setting Interval moving slides.

1. Setup Interval to run a function
2. Select element with active class
3. Remove active from current element
4. Select the next or restart the list of elements within the array
5. Add active to next slide

```
const arr = [];
const props = {
    "main": $(".slideArea"),
    "speed" : 3000
};

$(document).ready(function () {
    console.log('doc ready');
    makelistImages();
    outputImages();
    setInterval(moveSlides,props.speed);
})
```

```
function moveSlides(){
   const cur = $(".active");
   cur.removeClass('active');
   const next = cur.next();
   const newCur = next.length ? next : cur.prevAll().last();
   console.log(newCur);
   newCur.addClass('active');

}



function outputImages() {
   $.each(arr, function (index, value) {
       //console.log(value);
       let tempActive = index == 0 ? 'active' : '';
       //console.log(tempActive);
       let html = `<div class="slide ${tempActive}"><img
src='${value}'><span>Caption ${index+1}</span></div>`;
       props.main.append(html);
   })
}

function makelistImages() {
   for (let x = 0; x < 8; x++) {
       let temp =
`//via.placeholder.com/400x400/${rClr()}/${rClr()}?text=Image
${x+1}`;
```

```
        arr.push(temp);
    }
}

function rClr(){
    const temp = Math.random();
    //console.log(temp);
    const temp1 = temp.toString(16);
    //console.log(temp1);
    const temp2 = temp1.substr(2,6);
    //console.log(temp2);
    return temp2;
    // return Math.random().toString(16).substr(2,6);
}
```

Add Control Button Options for user event listeners and controls of slider.

Adding buttons for user interaction so that the user can control movement of slides to next and previous images. User interaction and event listeners with jQuery. Move to next slide on click.

Lesson Challenge

1. Add Control button options to screen
2. Apply CSS to buttons for control
3. Add Click events to the buttons
4. Create a function to move to next and then to move to previous slide.
5. Add events on click of button to move to new slide.

```
<!DOCTYPE html>
<html>
```

```
 <head><title>jQuery Slider Project</title>
 <link rel="stylesheet" href="style.css">
 </head>
 <body>
   <h1>jQuery Slider Project</h1>
   <div class="container">
    <div class="slideArea"></div>
    <div class="contBtns">
       <a class="prevBtn">Prev</a>
       <a class="nextBtn">Next</a>
    </div>
   </div>
   <script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.m
in.js"></script>
   <script src="app.js"></script>
 </body>
</html>

const arr = [];
const props = {
   "main": $(".slideArea"),
   "speed" : 3000
};

$(document).ready(function () {
   console.log('doc ready');
   makelistImages();
```

```
    outputImages();
    setInterval(moveSlides,props.speed);
})

$('.contBtns').on('click','.prevBtn',function(){
    console.log('prev');
    prevSlide();
})
$('.contBtns').on('click','.nextBtn',function(){
    console.log('next');
    nextSlide();
})

function nextSlide(){
    const cur = $(".active");
    cur.removeClass('active');
    const next = cur.next();
    const newCur = next.length ? next : cur.prevAll().last();
    console.log(newCur);
    newCur.addClass('active');
}
function prevSlide(){
    const cur = $(".active");
    cur.removeClass('active');
    const prev = cur.prev();
    const newCur = prev.length ? prev : cur.nextAll().last();
    console.log(newCur);
    newCur.addClass('active');
```

```
}



function moveSlides(){
   nextSlide();
}



function outputImages() {
   $.each(arr, function (index, value) {
       //console.log(value);
       let tempActive = index == 0 ? 'active' : '';
       //console.log(tempActive);
       let html = `<div class="slide ${tempActive}"><img
src='${value}'><span>Caption ${index+1}</span></div>`;
       props.main.append(html);
   })
}

function makelistImages() {
   for (let x = 0; x < 8; x++) {
       let temp =
`//via.placeholder.com/400x400/${rClr()}/${rClr()}?text=Image
${x+1}`;
       arr.push(temp);
   }
}
```

```
function rClr(){
   const temp = Math.random();
   //console.log(temp);
   const temp1 = temp.toString(16);
   //console.log(temp1);
   const temp2 = temp1.substr(2,6);
   //console.log(temp2);
   return temp2;
   // return Math.random().toString(16).substr(2,6);
}
*{
   box-sizing: border-box;
}
.contBtns{
   text-align: center;
   text-transform: uppercase;
}
.contBtns a{
   cursor: pointer;
   display: inline-block;
   padding:10px;
   border: 1px solid black;
   border-radius: 25px;
   margin:10px 0;
   width:100px;
   font-size: 1.2em;
   background-color:black;
```

```
    color:white;
}
.contBtns a:hover{
    background-color:blue;
}



.container{
    width:680px;
    margin:0 auto;
}
.slideArea {
    margin: 0 10px;
    position: relative;
    width:680px;
    height:500px;
    background-color: black;
}

.slide {
    position: absolute;
    opacity: 0;
    z-index: 10;
    padding:10px;
    text-align: center;
}
```

```
.slide img{

   width: 660px;
   height:480px;
   overflow: hidden;
}

.slide span{
   background-color: rgba(0,0,0,0.5);
   color:white;
   display: block;
   position: absolute;
   bottom:30px;
   width:100%;
   left:0px;
   padding:15px;
   text-transform: uppercase;
   font-size: 1.5em;
   overflow: hidden;

}

.active{
   opacity: 1;
   transition: opacity 1s ease;
   z-index: 11;
}
```

Update Intervals reset timer of interval on user interaction.

Debug the jQuery project update and reset the interval timer to restart with counters once a user interacts with the slider controls.

Lesson Challenge

1. Test of Slide project
2. Catch errors with intervals that should be reset after user interaction
3. Add interval as a property value to be able to clear the interval when needed.
4. Clean function of repeat coding
5. Add interval and clear interval functions.

```
const arr = [];
const props = {
   "main": $(".slideArea"),
   "speed" : 4000,
   "int" : {}
};
$(document).ready(function () {
   console.log('doc ready');
   makelistImages();
   outputImages();
   props.int = setInterval(moveSlides,props.speed);
})

$('.contBtns').on('click','.prevBtn',function(){
   console.log('prev');
   prevSlide();
   props.int = setInterval(moveSlides,props.speed);
```

```
})
$('.contBtns').on('click','.nextBtn',function(){
   console.log('next');
   nextSlide();
   props.int = setInterval(moveSlides,props.speed);
})

function updateSlideValue(newCur){
   clearInterval(props.int);
   newCur.addClass('active');
}

function nextSlide(){
   const cur = $(".active");
   cur.removeClass('active');
   const next = cur.next();
   const newCur = next.length ? next : cur.prevAll().last();
   updateSlideValue(newCur)

}
function prevSlide(){
   const cur = $(".active");
   cur.removeClass('active');
   const prev = cur.prev();
   const newCur = prev.length ? prev : cur.nextAll().last();
   updateSlideValue(newCur)
}
```

```
function moveSlides(){
   nextSlide();
}



function outputImages() {
   $.each(arr, function (index, value) {
       //console.log(value);
       let tempActive = index == 0 ? 'active' : '';
       //console.log(tempActive);
       let html = `<div class="slide ${tempActive}"><img
src='${value}'><span>Caption ${index+1}</span></div>`;
       props.main.append(html);
   })
}

function makelistImages() {
   for (let x = 0; x < 8; x++) {
       let temp =
`//via.placeholder.com/400x400/${rClr()}/${rClr()}?text=Image
${x+1}`;
       arr.push(temp);
   }
}

function rClr(){
```

```
    const temp = Math.random();
    //console.log(temp);
    const temp1 = temp.toString(16);
    //console.log(temp1);
    const temp2 = temp1.substr(2,6);
    //console.log(temp2);
    return temp2;
    // return Math.random().toString(16).substr(2,6);
}
```

jQuery slider Image Carousel Project Review code updates and debugging

Final code review and updates to clean the code and update styling.

```
<!DOCTYPE html>
<html>

<head>
 <title>jQuery Slider Project</title>
 <link rel="stylesheet" href="style.css">
</head>

<body>
 <h1>jQuery Slider Project</h1>
 <div class="container">
   <div class="slideArea"></div>
   <div class="contBtns">
```

```
      <a class="prevBtn">Prev</a>
      <a class="nextBtn">Next</a>
    </div>
  </div>
  <script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.m
in.js"></script>
  <script src="app.js"></script>
</body>

</html>
```

```
* {
    box-sizing: border-box;
}

.contBtns {
    text-align: center;
    text-transform: uppercase;
}

.contBtns a {
    cursor: pointer;
    display: inline-block;
    padding: 10px;
    border: 1px solid black;
    border-radius: 25px;
    margin: 10px 0;
```

```
    width: 100px;
    font-size: 1.2em;
    background-color: black;
    color: white;
}

.contBtns a:hover {
    background-color: blue;
}

.container {
    width: 90%;
    margin: 0 auto;
}

.slideArea {
    margin: 0 10px;
    position: relative;
    width: 100%;
    height: 500px;
    background-color: black;
}

.slide {
    position: absolute;
    opacity: 0;
    z-index: 10;
    padding: 10px;
```

```
    text-align: center;
    width: 100%;
}

.slide img {
    width: 100%;
    height: 480px;
    overflow: hidden;
}

.slide span {
    background-color: rgba(0, 0, 0, 0.5);
    color: white;
    display: block;
    position: absolute;
    bottom: 30px;
    width: 100%;
    left: 0px;
    padding: 15px;
    text-transform: uppercase;
    font-size: 1.5em;
    overflow: hidden;
}

.active {
    opacity: 1;
    transition: opacity 1s ease;
    z-index: 11;
```

```
}
```

```
const arr = [];
const props = {
   "main": $(".slideArea"),
   "speed": 1000,
   "int": {}
};

$(document).ready(function () {
   makelistImages();
   outputImages();
   props.int = setInterval(nextSlide, props.speed);
})

$('.contBtns').on('click', '.prevBtn', function () {
   prevSlide();
   props.int = setInterval(nextSlide, props.speed);
})
$('.contBtns').on('click', '.nextBtn', function () {
   nextSlide();
   props.int = setInterval(nextSlide, props.speed);
})

function updateSlideValue(newCur) {
   clearInterval(props.int);
   newCur.addClass('active');
}
```

```
function nextSlide() {
   const cur = $(".active");
   const next = cur.removeClass('active').next();
   const newCur = next.length ? next : cur.prevAll().last();
   updateSlideValue(newCur);

}

function prevSlide() {
   const cur = $(".active");
   const prev = cur.removeClass('active').prev();
   const newCur = prev.length ? prev : cur.nextAll().last();
   updateSlideValue(newCur);
}

function outputImages() {
   $.each(arr, function (index, value) {
       let tempActive = index == 0 ? 'active' : '';
       let html = `<div class="slide ${tempActive}"><img
src='${value}'><span>Caption ${index+1}</span></div>`;
       props.main.append(html);
   })
}

function makelistImages() {
   for (let x = 0; x < 9; x++) {
```

```
        let temp =
`//via.placeholder.com/400x400/${rClr()}/${rClr()}?text=Image
${x+1}`;
        arr.push(temp);
    }
}

function rClr() {
    return Math.random().toString(16).substr(2, 6);
}
```