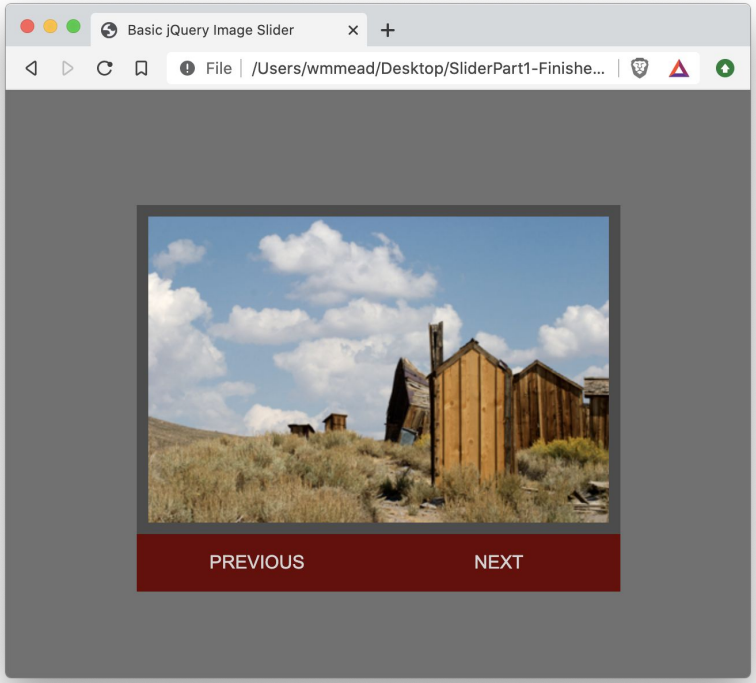
jQuery Image Slider

Version 1 - Basic Slider

Final Project

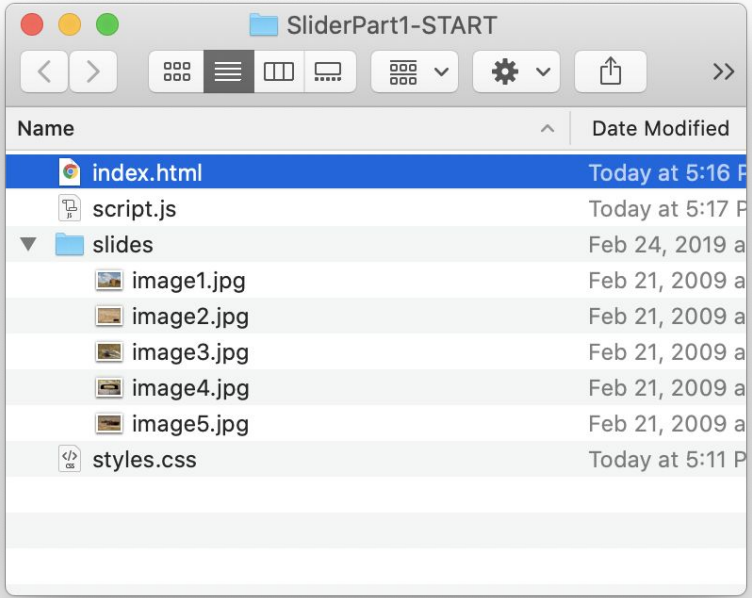
At the end, you will get a simple jQuery images slider that you can put on any web page.



Start Folder

The start folder includes the basic HTML file, the CSS file and an empty script file, where you can add the jQuery needed to make this project work.

There is a slides folder with slides for each image.



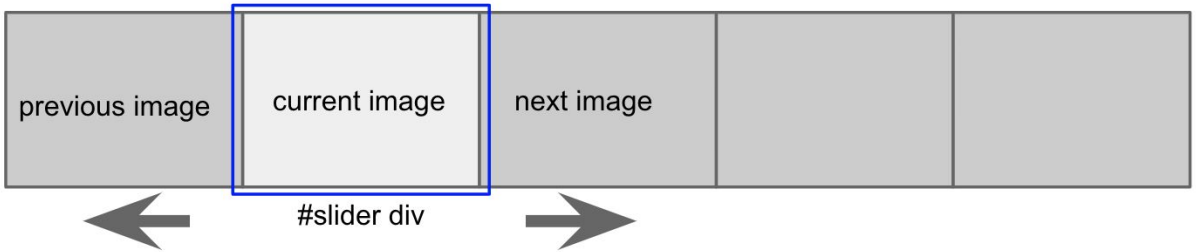
HTML for the Project

```
<div id="slider">
    <ul>
        <li><img src="slides/image1.jpg" alt="slideshow image"></li>
        <li><img src="slides/image2.jpg" alt="slideshow image"></li>
        <li><img src="slides/image3.jpg" alt="slideshow image"></li>
        <li><img src="slides/image4.jpg" alt="slideshow image"></li>
        <li><img src="slides/image5.jpg" alt="slideshow image"></li>
    </ul>
</div>

<p id="links"><a href="#" id="previous">previous</a><a href="#" id="next">next</a></p>
```

All the HTML you need for this project is in the index.html file.

Project Strategy



The basic strategy is that the slides are inside an element with the ID set to “slider” That element is set to the width and height of one image in the slider, then set to `position: relative`.

Check Out the Styles

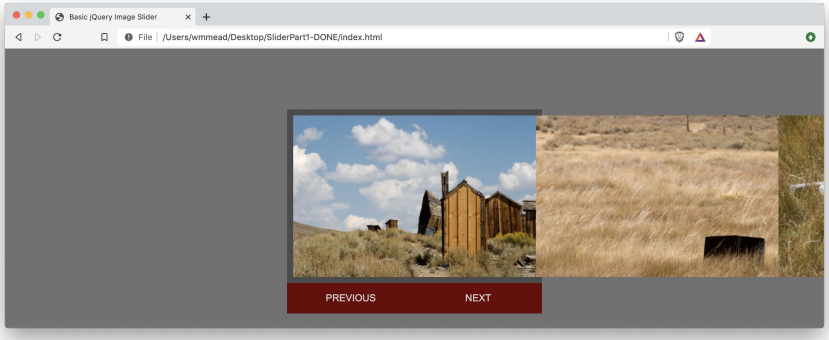
The important bits on these rules are highlighted here. The parent #slider is set to the correct width and height for the images and has overflow: hidden, so you won't see the slides that are outside of the container.

The slides are absolutely positioned and pinned to the upper left corner of the parent window. You will be animating the "left" property to create the slider.

```
#slider {
    width:400px;
    overflow:hidden;
    border: 10px solid #4C4C4C;
    height:266px;
    position:relative;
    margin:auto;
}

#slider ul {
    list-style-type: none;
    margin: 0;
    padding: 0;
    position: absolute;
    top: 0;
    left: 0;
    display: flex;
}
```

Images Lined Up




```
#slider {
    width:400px;
    /*overflow:hidden;*/
    border: 10px solid #4C4C4C;
    height:266px;
    position:relative;
    margin:auto;
}
```

If you turn off overflow: hidden, in the #slider rule just for a minute, you will see that the other images are lined up next to the first image in the window.

Making the Script General

```
const imageCount = $("#slider ul li").length;
const imageWidth = $("#slider ul li:first img").width();
alert(imageWidth);
```

The script should be as general as possible. The first line above tells you how many images are in the slider, and the second tells you how wide the first images is.

Window Load Function

```
$(window).on('load', function () {
      "use strict";

      const imageCount = $("#slider ul li").length;
      const imageWidth = $("#slider ul li:first img").width();
      alert(imageWidth);

});
```

Use the jQuery .on() method and move the code so that it is inside the anonymous callback function. This method will run the callback function when all the images have loaded.

It also acts as a closure, so you can just add the “use strict” directive.

Add a Few More Variables

Next, add a few more variables. Total width is the width of the first image times the number of images. This should alert out to be 2000px.

Then the leftPosition variable will change based on which slide is showing. It's 0 for the first slide, and -400px for the second slide, etc..

The counter will help us keep track of which slide we are on.

```
$(window).on('load', function () {
      "use strict";

      const imageCount = $("#slider ul li").length;
      const imageWidth = $("#slider ul li:first img").width();
      const totalWidth = (imageCount * imageWidth) + "px";
      alert(totalWidth);

      let leftPosition = 0;
      let counter = 0;

      $("#slider ul").css("width", totalWidth);

});
```

Add Click Handler for “Next” Button

Add a click handler for the next link, and inside it, increment the counter.

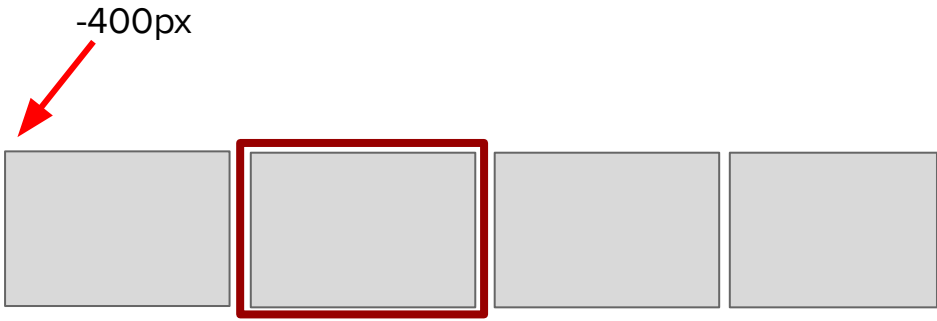
Then set the left position to a negative number that ends with “px”.

When the link is clicked the first time, leftPosition will become -400px, which slides the whole strip of images to the left 400px, putting the second image in the window.

```
$("#next").click( function(){

    counter++;
    leftPosition = `-${counter * imageWidth}px`;

} );
```



Animate the Slide Strip

```
$("#next").click( function(){

    counter++;
    leftPosition = `-${counter * imageWidth}px`;
    $("#slider ul").animate( {left : leftPosition}, 700, "easeInQuad" );

} );
```

Now, animate the slide strip using the custom animate method. The easing plugin has already been linked so you can use the “easeInQuad” easing setting.

This will work, until you get to the end of the strip of slides.

Next Click Handler with If Statement

```
$("#next").click( function(){

    counter++;

    if( counter == imageCount ){
        counter = 0;
    }

    leftPosition = `-${counter * imageWidth}px`;

    $("#slider ul").animate( {left : leftPosition}, 700, "easeInQuad" );

} );
```

Here is the click handler for the next button. It should make sense, given the previous slide shows you have done already.

Can you do the previous click handler by yourself?

Previous Image Click Handler

```
$("#previous").click( function(){

    counter--;

    if( counter < 0 ){
        counter = imageCount-1;
    }

    leftPosition = `-${counter * imageWidth}px`;

    $("#slider ul").animate( {left : leftPosition}, 700, "easeInQuad" );

} );
```

Did you get something like this?

Summary

You were able to get this image slider to work using jQuery and the built in effects library.

It may seem weird that the slides go all the way back to the beginning, when you get to the end.

In the second version you will fix that.