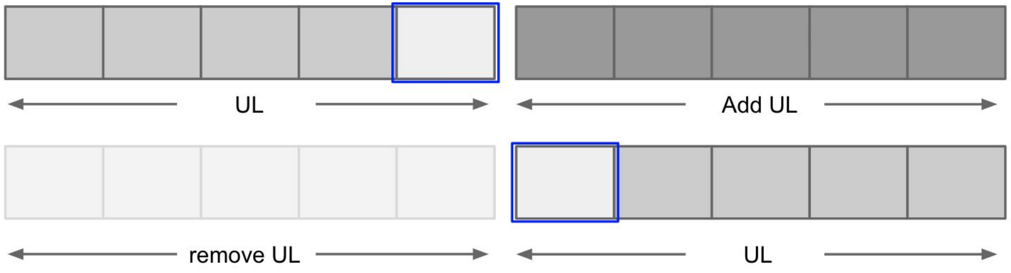
jQuery Image Slider

Version 2

Strategy



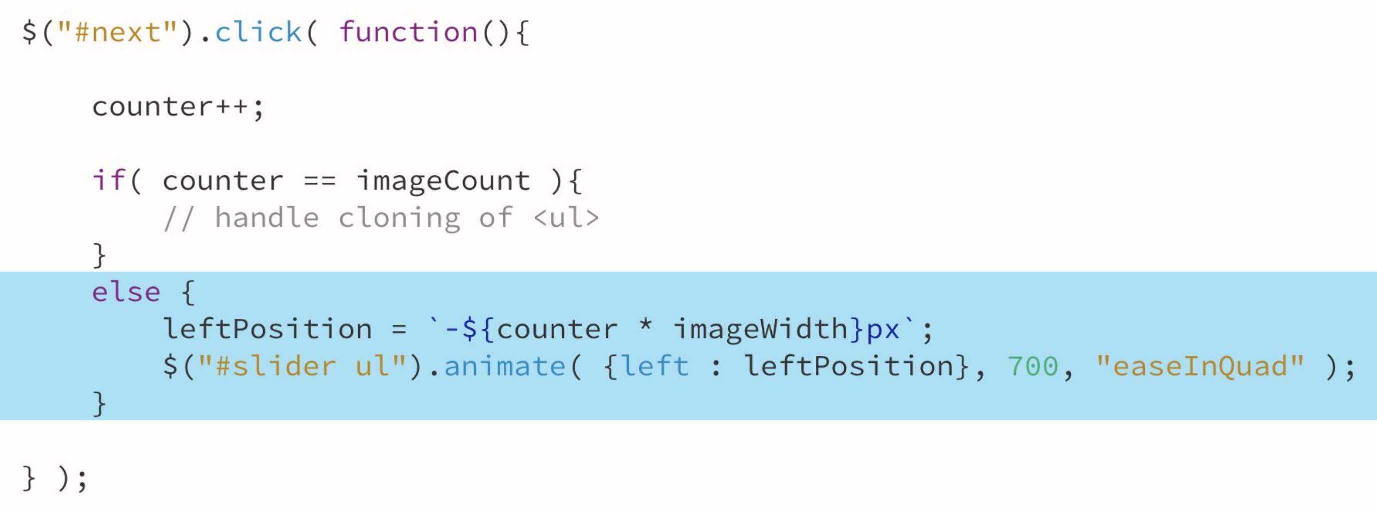
Start with the finished files for version 1 of the slider. From here, you will modify the script to make it look like the slides are looping.

To do that, when the user gets to the end of the strip, clone the strip and stick it on the end.

Then, when the cloned version of the strip slides into place, remove the original strip.

For a split second, there will be two unordered lists inside the #slider div.

Start Work on the Next Click Handler



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

    counter++;

    if( counter == imageCount ){
        // handle cloning of <ul>
    }
    else {
        leftPosition = `-${counter * imageWidth}px`;
        $("#slider ul").animate( {left : leftPosition}, 700, "easeInQuad" );
    }

} );
```

For the next button click handler, add an else statement and put the two lines from the bottom of the previous version inside the else statement.

This will handle the normal progression of slides, assuming you are not at the end of the strip of slides. Everything else will happen in the if statement.

Cloning the Unordered List

```
if (counter == imageCount) {

      $("#slider ul").clone().appendTo("#slider");
      $("#slider ul").last().css("left", imageWidth + "px");

}
```

Focusing on the if statement, the first thing to do is to clone the unordered list, and then append it to the slider div.

This will put a second unordered list in the slider div after the first unordered list.

This second list, (the last one), will have it’s left position set to the width of one of the images, which puts it up at the right hand corner of the slider div window.

Setting the Left Position

```
if (counter == imageCount) {

      $("#slider ul").clone().appendTo("#slider");
      $("#slider ul").last().css("left", imageWidth + "px");

      leftPosition = `-${totalWidth}`;

}
```

Set the leftPostion variable to totalWidth with a minus sign in front of it. This will make it -2000px, which will be used to animate the first strip of images so that they are just off the left side of the slider div.

After this, two things will happen at the same time. The original strip of slides will slide to leftPosition, then the new strip of slides will slide to zero.

Animating Both Slide Strips

```
leftPosition = `-${totalWidth}`;

$("#slider ul").last().animate({ left: 0 }, 700, "easeInQuad");
$("#slider ul").first().animate({ left: leftPosition }, 700, "easeInQuad");
```

Still working inside the if statement, you will animate BOTH slide strips. Even though these statements come one after another, the animations will happen at the exact same time.

The new strip of slides gets animated to position zero, putting the first slide in the window.

The old strip gets animated off the left side of the slider window.

Add a Callback Function

```
leftPosition = `-${totalWidth}`;

$("#slider ul").last().animate({ left: 0 }, 700, "easeInQuad");
$("#slider ul").first().animate({ left: leftPosition }, 700, "easeInQuad", function () {
      $("#slider ul").first().remove();
});
```

After the first strip animates off the side of the slider window, you need to remove it from the DOM entirely.

This can be done with a callback function. After that animation is complete, remove it!

Finally, Set the Counter to 0

```
$("#next").click(function () {
      counter++;
      if (counter == imageCount) {
            $("#slider ul").clone().appendTo("#slider");
            $("#slider ul").last().css("left", imageWidth + "px");

            leftPosition = `-${totalWidth}`;

            $("#slider ul").last().animate({ left: 0 }, 700, "easeInQuad");
            $("#slider ul").first().animate({ left: leftPosition }, 700, "easeInQuad", function () {
                  $("#slider ul").first().remove();
            });
            counter = 0;
      } else {
            leftPosition = `-${counter * imageWidth}px`;
            $("#slider ul").animate({ left: leftPosition }, 700, "easeInQuad");
      }
});
```

Here is the entire click handler. Don’t forget to set the counter back to 0.

Can you do the one for the previous button yourself?

Previous Button Click Handler

```
$("#previous").click(function () {
      counter--;
      if (counter < 0) {
            counter = imageCount - 1;

            $("#slider ul").clone().appendTo("#slider");
            $("#slider ul").last().css("left", `-${totalWidth}`);
            leftPosition = `-${counter * imageWidth}px`;
            $("#slider ul").last().animate({ left: leftPosition }, 700, "easeInQuad");
            $("#slider ul").first().animate({ left: imageWidth + "px" }, 700, "easeInQuad", function () {
                  $("#slider ul").first().remove();
            });
      } else {
            leftPosition = `-${counter * imageWidth}px`;
            $("#slider ul").animate({ left: leftPosition }, 700, "easeInQuad");
      }
});
```

Summary

On the previous slide is the finished previous click handler. See if you can make sense of it. Much of it is the same as the next click handler, just backwards.

This is a tough script to wrap your head around, because there is so much DOM manipulation going on.