JavaScript

JavaScript Image Spinner – Step by Step Instructions

Notes on these Instructions

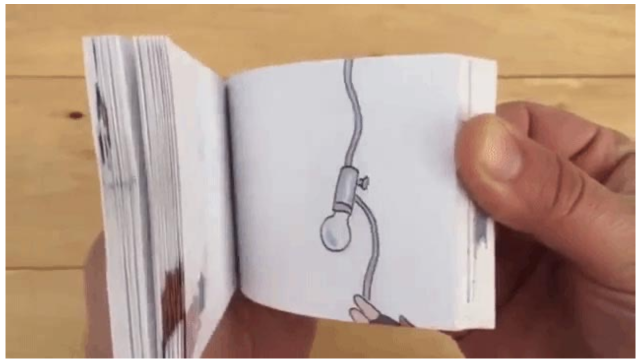
- These instructions are not the only way to complete this assignment
- Usually JavaScript provides multiple ways to complete the same task
- If you find an alternative method to creating this image spinner,
  I encourage you to explore it and give it a try

What We Are Going to Build

- According to the a-02 instruction sheet we need to:
  - Create a script file and attach it to the index.html file
  - Write the JavaScript that will rotate the image clockwise when we "click" the "< Turn" button
  - Write the JavaScript that will rotate the image counter-clockwise when we "click" the "Turn >" button

How This Spinner Works

- The "3D" rotation of the image is a bit of an illusion created with a similar technique from that used in "flip-book" animation books that you flip through quickly to create the illusion of motion

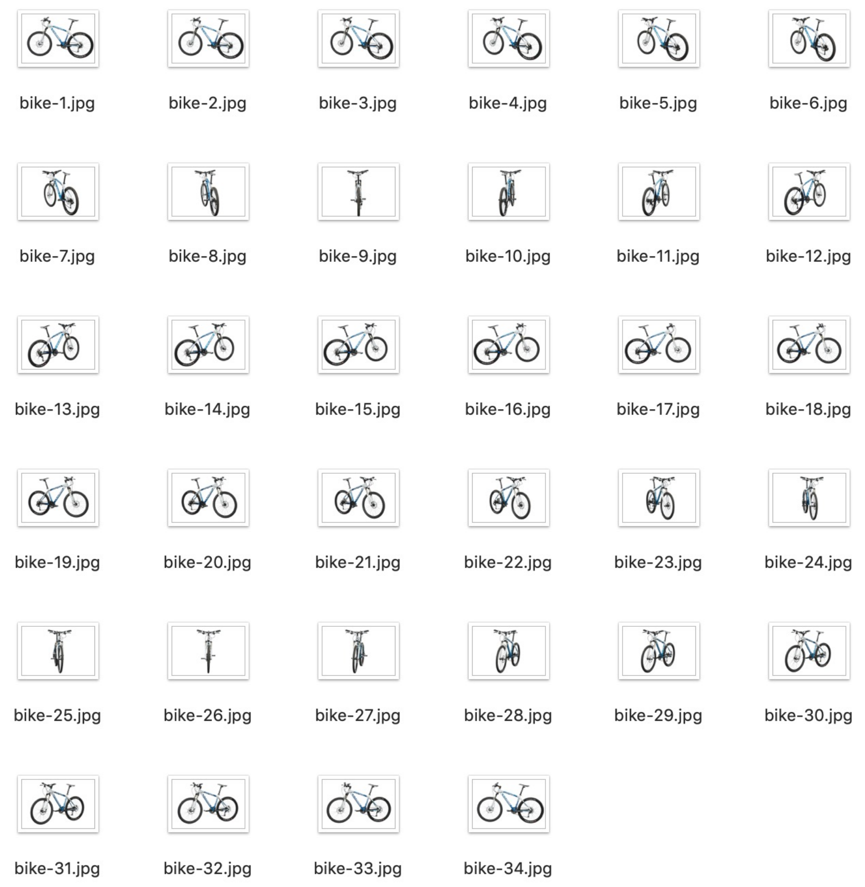


- Click the link below to see this flip book animating
  - https://laughingsquid.com/flipbook-every-booby-trap-in-home-alone/

Images from: https://laughingsquid.com/flipbook-every-booby-trap-in-home-alone/

How This Spinner Works

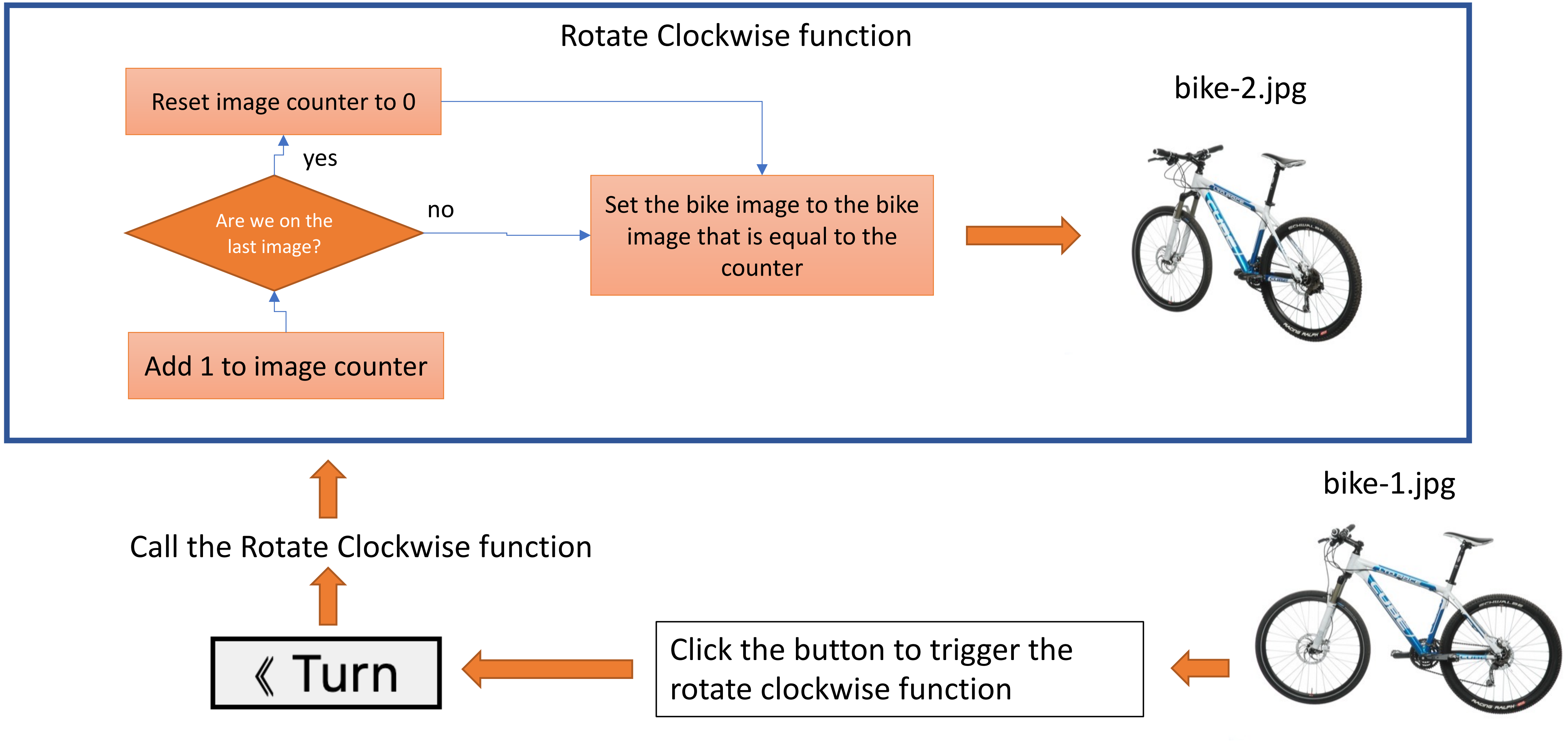
- For our bike spinner we use 34 single images of the bike
  - The images are stored in the images folder and named:
    - bike-1.jpg, bike-2.jpg, bike-3.jpg…
- Each of the 34 images is a photo of the bike at a different degree of rotation



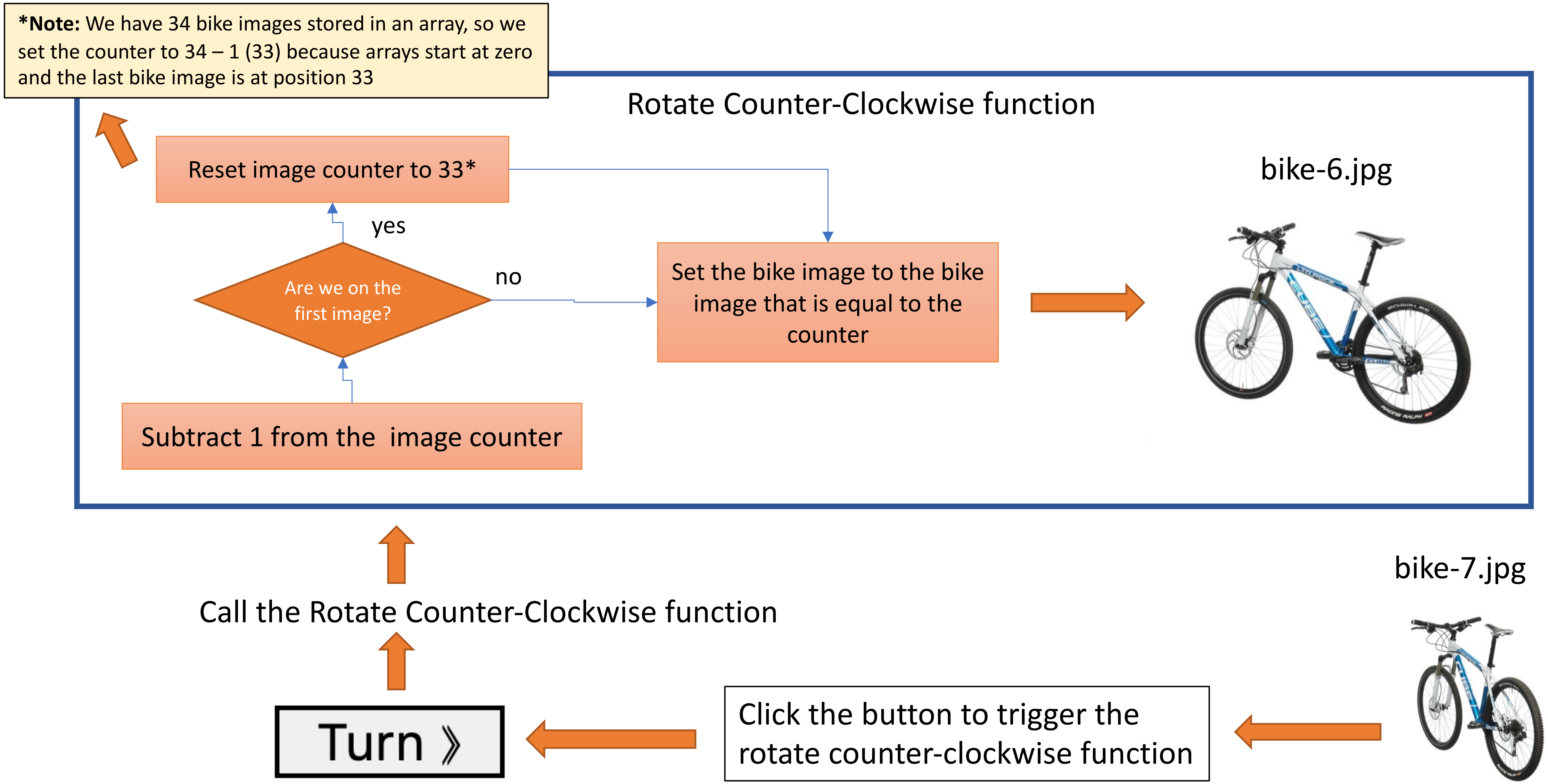
How This Spinner Works

- To create the illusion of the bike rotating we will write functions that move through the images sequentially
    - Rotate Clockwise function
        - Increments images - (1, 2, 3…)
    - Rotate Counter-Clockwise function
        - Decrements images - (7, 6, 5…)
- These functions will be triggered when we click our "< Turn" and "Turn >" buttons

How this Spinner Works – Rotate Clockwise


Rotate Clockwise function
Reset image counter to 0
yes
Are we on the last image?
no
Set the bike image to the bike image that is equal to the counter
bike-2.jpg
Add 1 to image counter
Call the Rotate Clockwise function
bike-1.jpg
《 Turn
Click the button to trigger the rotate clockwise function

How this Spinner Works – Rotate Counter-Clockwise



***Note:** We have 34 bike images stored in an array, so we set the counter to 34 – 1 (33) because arrays start at zero and the last bike image is at position 33

Rotate Counter-Clockwise function
Reset image counter to 33*
yes
Are we on the first image?
no
Set the bike image to the bike image that is equal to the counter
Subtract 1 from the image counter
bike-6.jpg
Call the Rotate Counter-Clockwise function
Turn ›
Click the button to trigger the rotate counter-clockwise function
bike-7.jpg

The Code

For the Bike Spinner

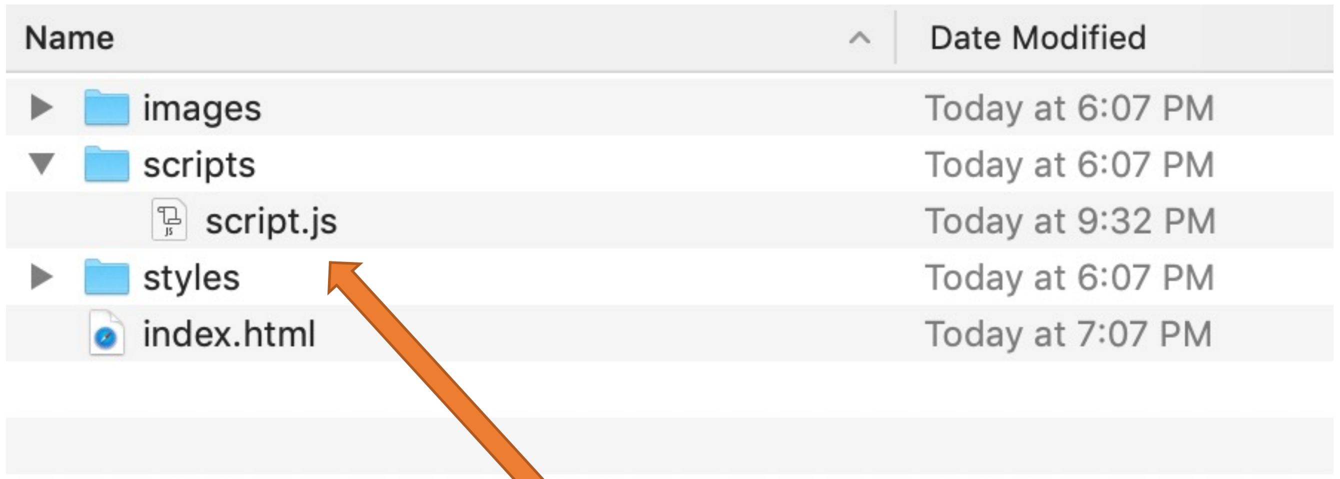
Step 1

Creating the script.js File

Create the Script File

- Create a JavaScript file called "script.js"
- Save this file inside the "scripts" folder

a-02-start folder



Name	Date Modified
▶ images	Today at 6:07 PM
▼ scripts	Today at 6:07 PM
script.js	Today at 9:32 PM
▶ styles	Today at 6:07 PM
index.html	Today at 7:07 PM

Create a new "script.js" file and store it in the "scripts" folder

Step 2

Attaching the script.js File to the HTML Document

Attaching the JS File to the HTML Document

- Attach the newly created "script.js" file to the "index.html" document by using a script tag placed immediately before the closing "body" element

index.html file

```html
<script src="scripts/script.js"></script>
</body>
</html>
```

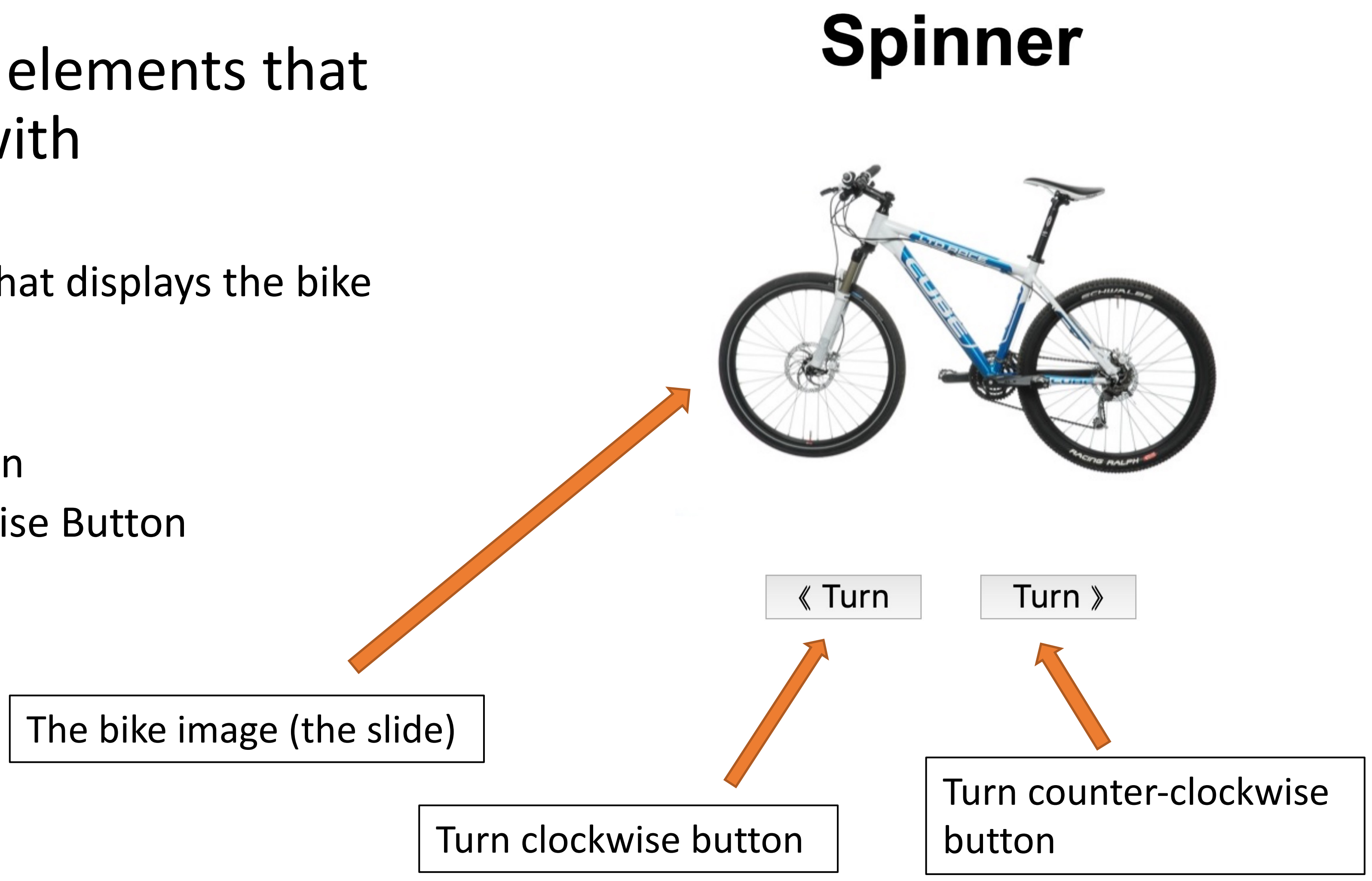
Use the "script" element with a "src" attribute to attach an external script file to an HTML document

Step 3

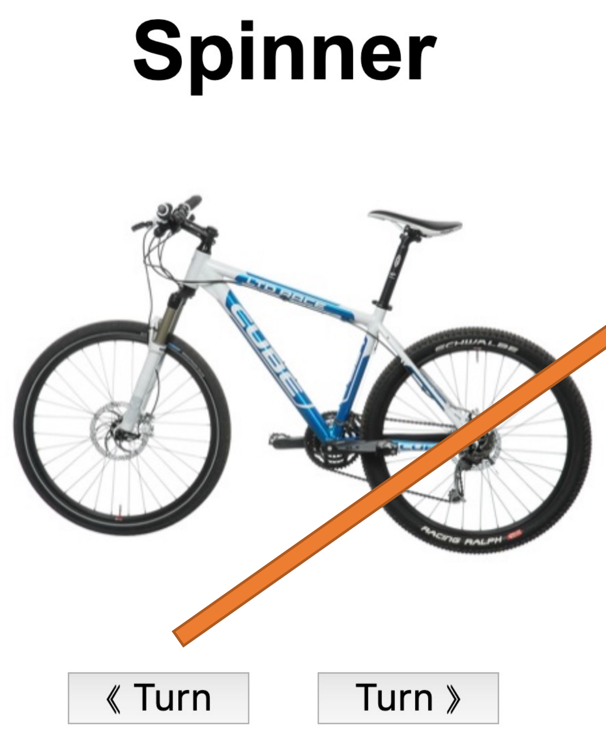
Grabbing the HTML Elements

Grabbing the HTML Elements

- Determine the HTML elements that we need to interact with
    - Slide
        - The image element that displays the bike image
    - Buttons
        - Turn Clockwise Button
        - Turn Counter-Clockwise Button



Grabbing the HTML Elements – The Clockwise Button



The Turn Clockwise button

The HTML for the Turn Clockwise Button

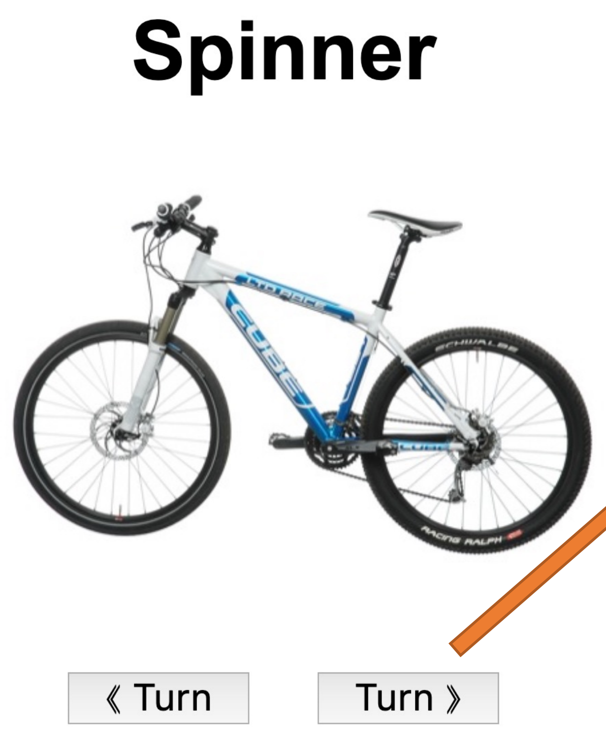
```
<button id="btn-turn-clockwise">&Lang; Turn</button>
```

Select this element with JavaScript and store it in a variable called "btnTurnCW"

JS (place this code in the "script.js" file)

```
const btnTurnCW = document.getElementById('btn-turn-clockwise');
```

Grabbing the HTML Elements – The Counter-Clockwise Button



The Turn Counter-Clockwise button

The HTML for the Turn Counter-Clockwise Button

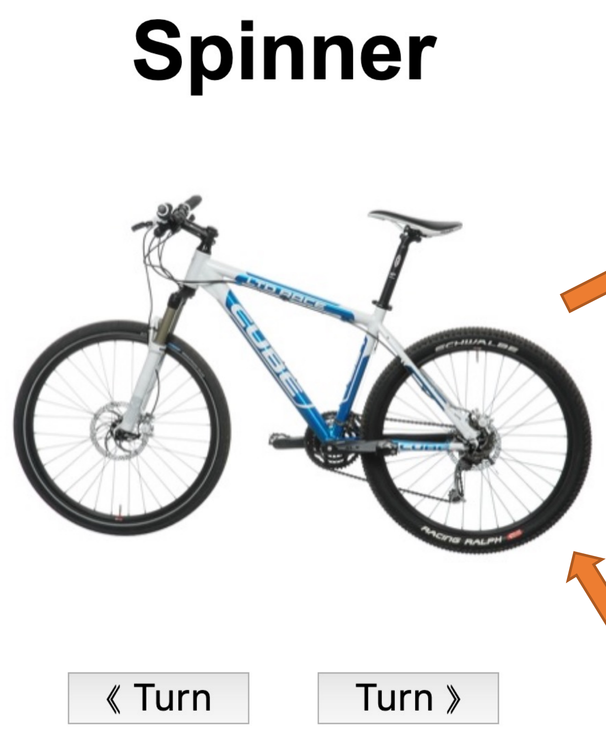
```html
<button id="btn-turn-counter-clockwise">Turn &Rang;</button>
```

Select this element with JavaScript and store it in a variable called "btnTurnCCW"

JS (place this code in the "script.js" file)

```js
const btnTurnCCW = document.getElementById('btn-turn-counter-clockwise');
```

Grabbing the HTML Elements – The Slide



The bike image (the slide)

The HTML for the Bike Slide

```
<img id="slide" src="images/bike-1.jpg" alt="Bike">
```

Select this element with JavaScript and store it in a variable called "slide"

JS (place this code in the "script.js" file)

```
const slide = document.getElementById('slide');
```

Grabbing the HTML Elements – The Complete JS Code

Add the following code to the top of the script.js file

script.js

```
// Button Elements
const btnTurnCW = document.getElementById('btn-turn-clockwise');
const btnTurnCCW = document.getElementById('btn-turn-counter-clockwise');

// HTML Slide Element (the bike image element)
const slide = document.getElementById('slide');
```

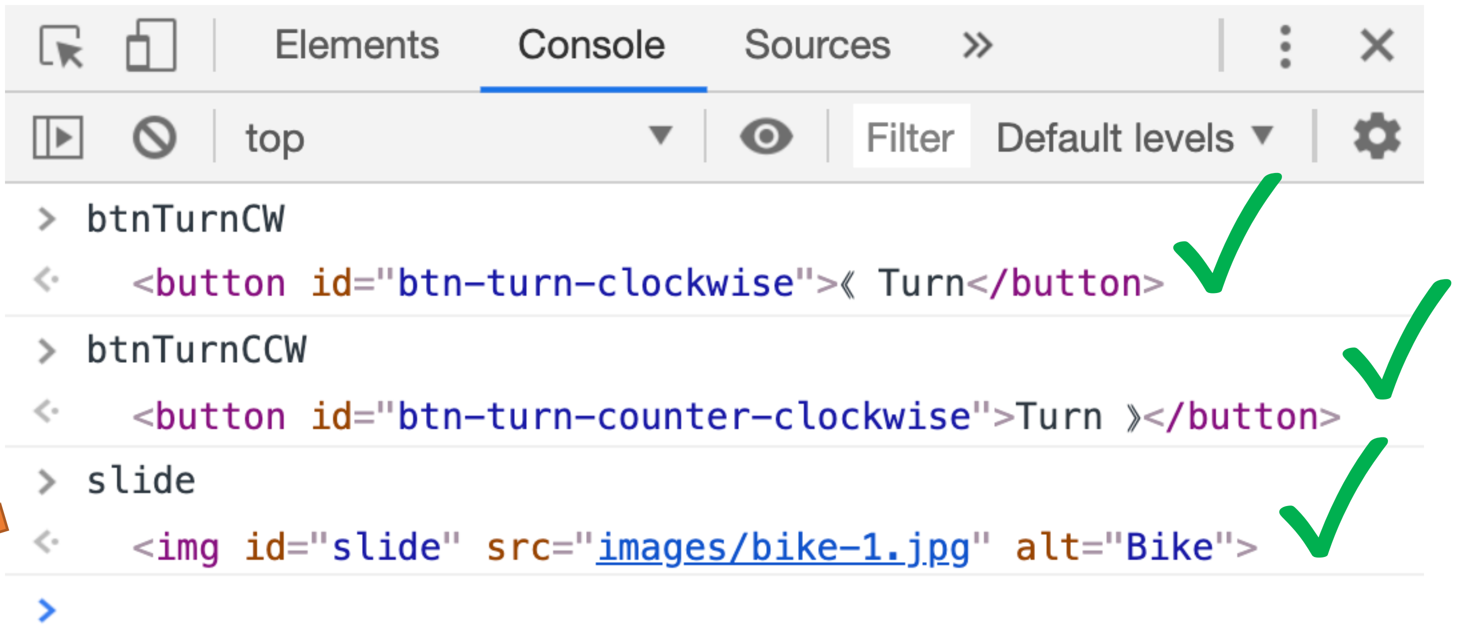
Pause and Check Your Code

Check your code for errors before proceeding

- Check early and check often when writing a script
- Test to make sure your buttons and the slide have been selected correctly
  - Test by entering your variable names into the console
  - If your variables return HTML elements you are good to go
  - If your variables return "null":
    - Check your code, and make sure the ID in the HTML match the ID in the "document.getElementById()" method

Check your code for errors before proceeding

All our HTML element variables return HTML elements, so we are good to go!



```
> btnTurnCW
<·   <button id="btn-turn-clockwise">《 Turn</button>
> btnTurnCCW
<·   <button id="btn-turn-counter-clockwise">Turn 》</button>
> slide
<·   <img id="slide" src="images/bike-1.jpg" alt="Bike">
>
```

Step 4

Set the Slide Variables

Slide Variables

- Our slide variables will store the following
  - numberOfSlides
    - The number of slides we will be using in this slide show
      - 34 -> the number of bike images
  - counter
    - Used to keep track of the current slide being displayed in the HTML document
    - This is based on the position of the image in our slides array (more on that later)
    - Since the counter is based on an array we initialize it with a starting value of zero

Slide Variables – The Complete JS Code

Add the following code to the script.js file after the HTML element variables

script.js

```
// Number of slides
const numberOfSlides = 34;

// Counter - used to keep track of the current slide
// *** We start out at slide 1, which is position
// zero in our slides array (set later on)
let counter = 0;
```

Step 5

Create our Array of Bike Images

Creating our Array of Bike Images

- We need to create an array of 34 bike images
- We could hand code 34 bike images into an array, but instead we will have JavaScript do the monotonous work for us by utilizing a loop

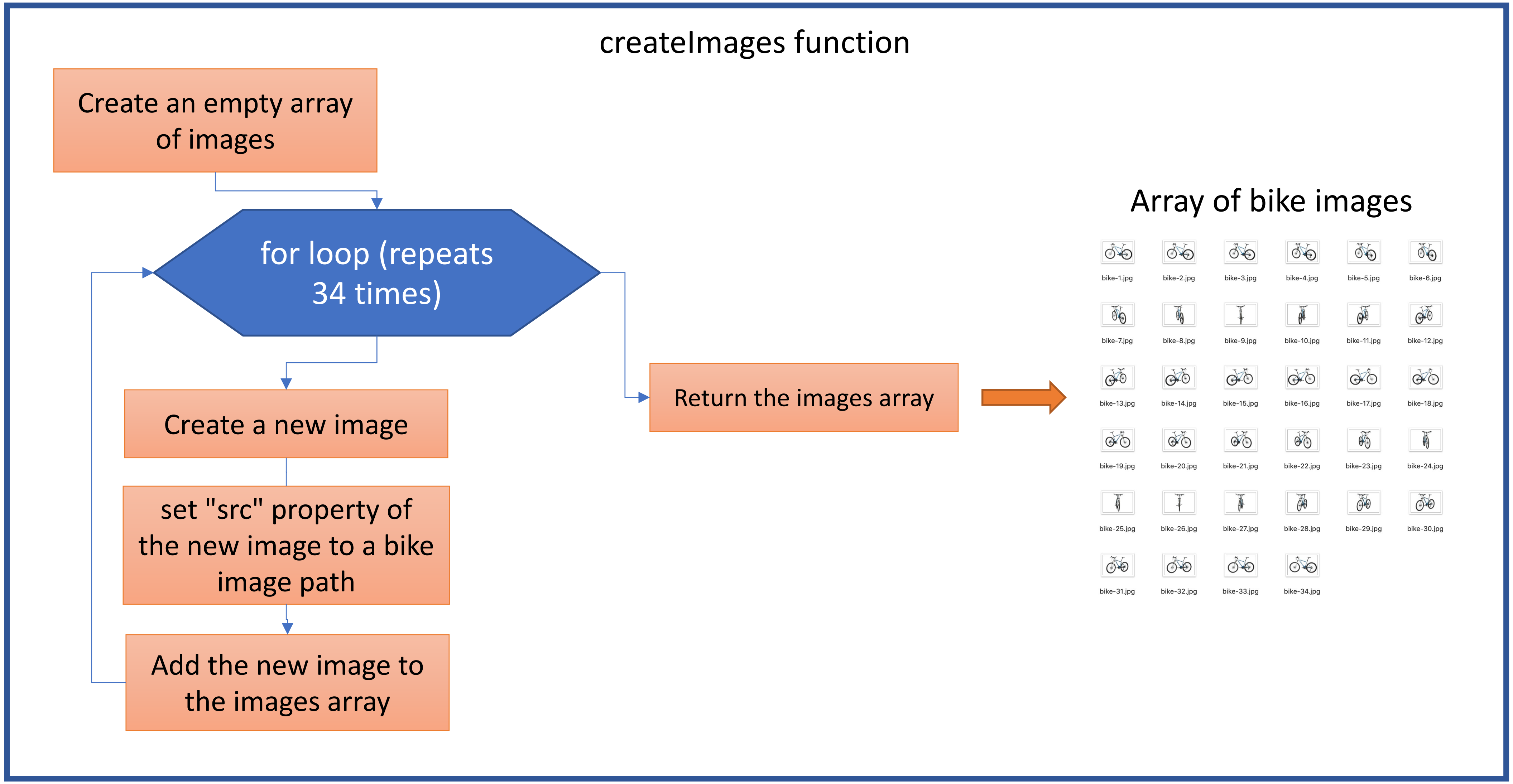
createImages() Function

- To keep our code clean and organized we will create a "createImages()" function that will take two parameters
  - numberOfSlides
    - This sets the number of images we will need to create and add to our slides array
    - This will be set to 34 when we call this function
  - imageName
    - This sets the core image name
      - This will be set to "bike" when we call this function

createImages() Function

- The createImages() function will first create an empty array which we will eventually fill with bike images
- The function will then run a loop which does the following
  - Creates a new image using JavaScript's built-in new Image() constructor function
  - Sets the "src" property of the newly created image to a file path that points to a bike image
    - Example: "images/bike-1.jpg", "images/bike-2.jpg", "images/bike-3.jpg"…
  - Inserts the newly created image at the end of the array
- Finally, our function will "return" the array of images

createImages() function



createImages() Function Code

script.js

```
// createImages
// - Creates an array of images
function createImages(numberOfImages, imageName){

   const imageList = [];

   // Use a loop to generate the array of
   // 34 bike images
   for(let i = 1; i <= numberOfImages; i++){
      const img = new Image();
      // Set the "src" property of the newly
      // created image object
      img.src = `images/${imageName}-${i}.jpg`;
      // Append the new image to the
      // imageList
      imageList.push(img);
   }

   return imageList;

} // end createImages
```

Add the following code to the end of the script.js file

Call the createImages() Function

- With our createImages() function created we now need to call it and pass in the values needed for our slide show
- We store the returned images array in a variable

Call the createImages() Function - Code

Add the following code to script.js after the slide variables code

script.js

```
// Create Slides
const slides = createImages(numberOfSlides, 'bike');
```

Store the returned array of images in a variable

Pass in our numberOfSlides (34) variable into the function

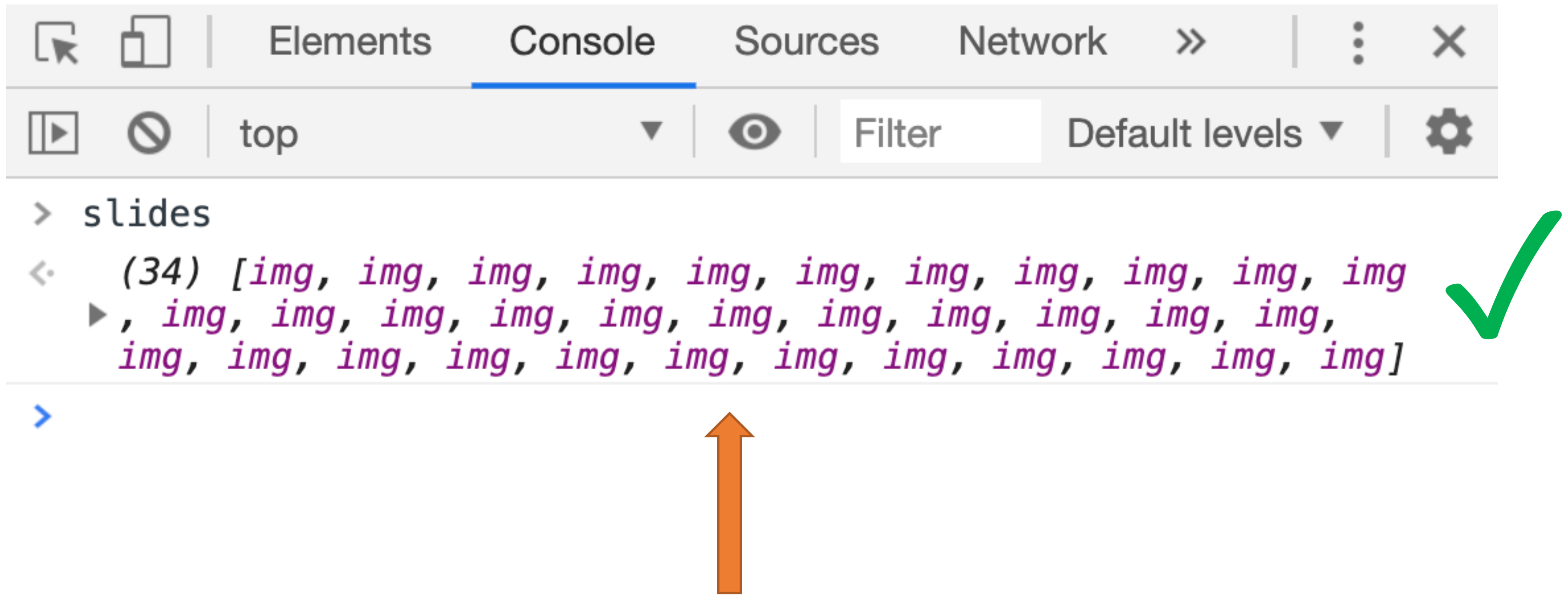
Pass in our image name (bike) into the function

Pause and Check Your Code

Check your code for errors before proceeding

- Check your code often when writing a script
- We created and called a function, we should test it to see if it works before proceeding
- We should test if our function returned an array of bike images
  - Test by entering the slides variable name into the console
  - If your variable returns an array of 34 bike images you are probably good to go
  - If your variables returns an empty array or another value, check your code

Check your code for errors before proceeding



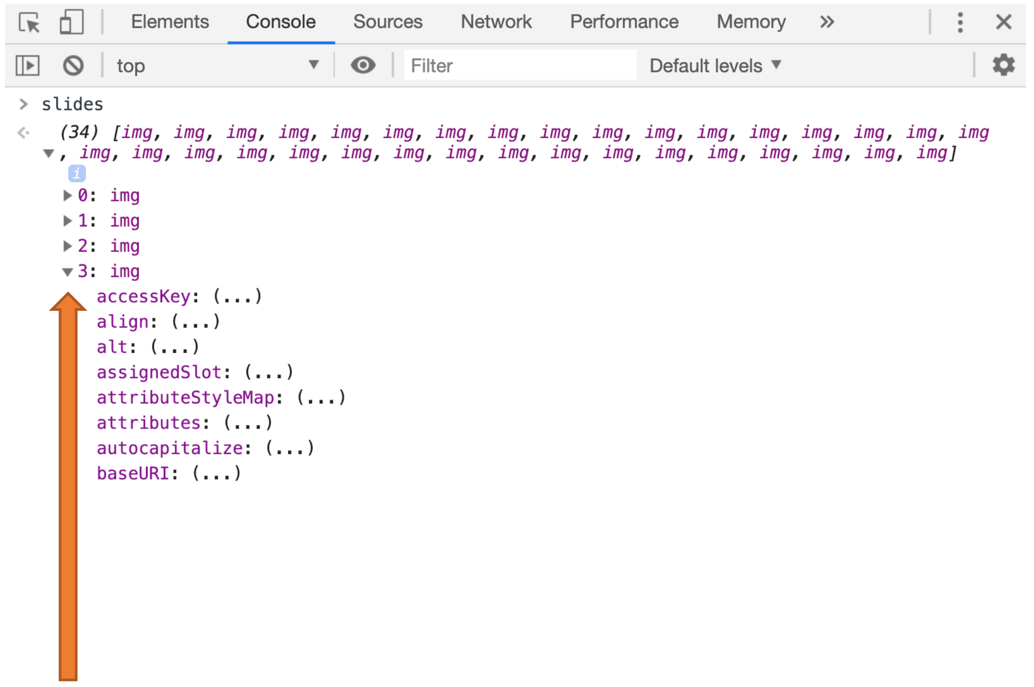
```
> slides
<· (34) [img, img, img, img, img, img, img, img, img, img, img
 ▶ , img, img, img, img, img, img, img, img, img, img, img,
   img, img, img, img, img, img, img, img, img, img, img, img]
>
```

Our slides variable returns an array of images, so we are probably good to go!

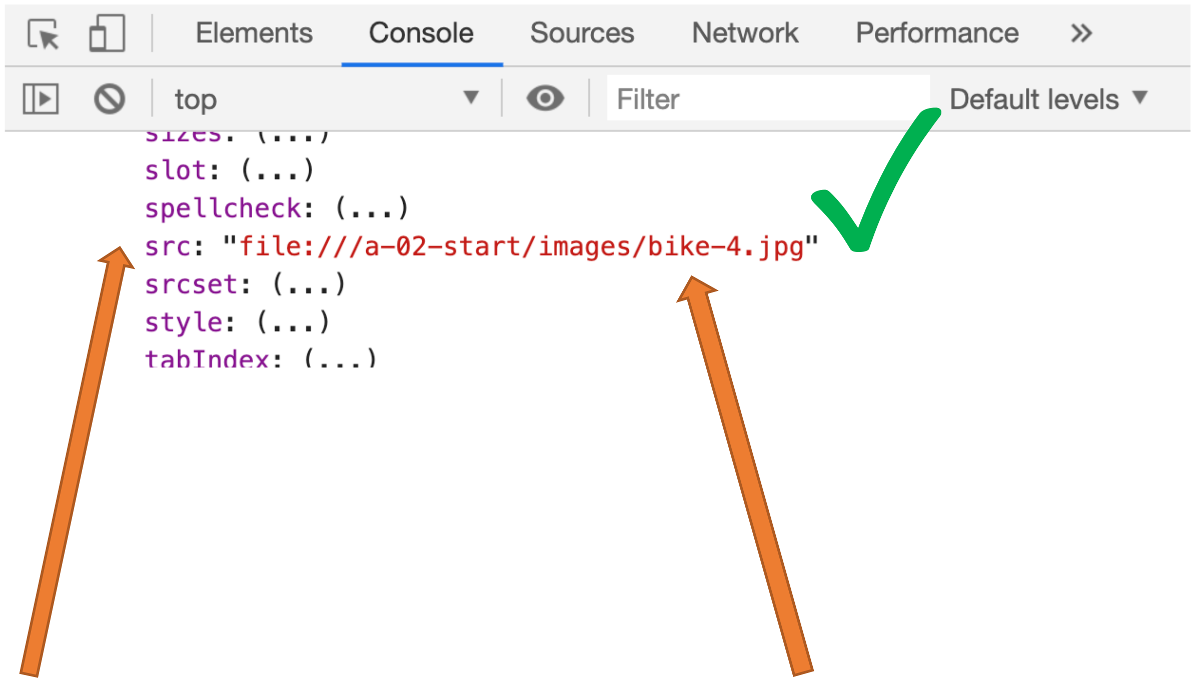
Check your code for errors before proceeding

- In the previous slide I said we are "probably" ok
- To be 100% sure, we should check to make sure that our images contained inside our "slides" array have "src" properties that point to our bike images
- We can check this in the console by opening one of the image items in the array and checking it's "src" property
  - See the next slide for details

Check your code for errors before proceeding



```
> slides
< (34) [img, img, img, img, img, img, img, img, img, img, img, img, img, img, img, img, img
▼ , img, img, img, img, img, img, img, img, img, img, img, img, img, img, img, img, img]
  ▶0: img
  ▶1: img
  ▶2: img
  ▼3: img
     accessKey: (...)
     align: (...)
     alt: (...)
     assignedSlot: (...)
     attributeStyleMap: (...)
     attributes: (...)
     autocapitalize: (...)
     baseURI: (...)
```



```
sizes: (...)
slot: (...)
spellcheck: (...)
src: "file:///a-02-start/images/bike-4.jpg"
srcset: (...)
style: (...)
tabIndex: (...)
```

Click the arrow on the slides array to crack open the array. Then click on the down arrow of any one of the "img" array items

Scroll through the long list of properties and methods until you find the "src" property. Check to make sure it points to one of the bike images

Looks like our "src" points to one of our bike images, so we are now good to go!

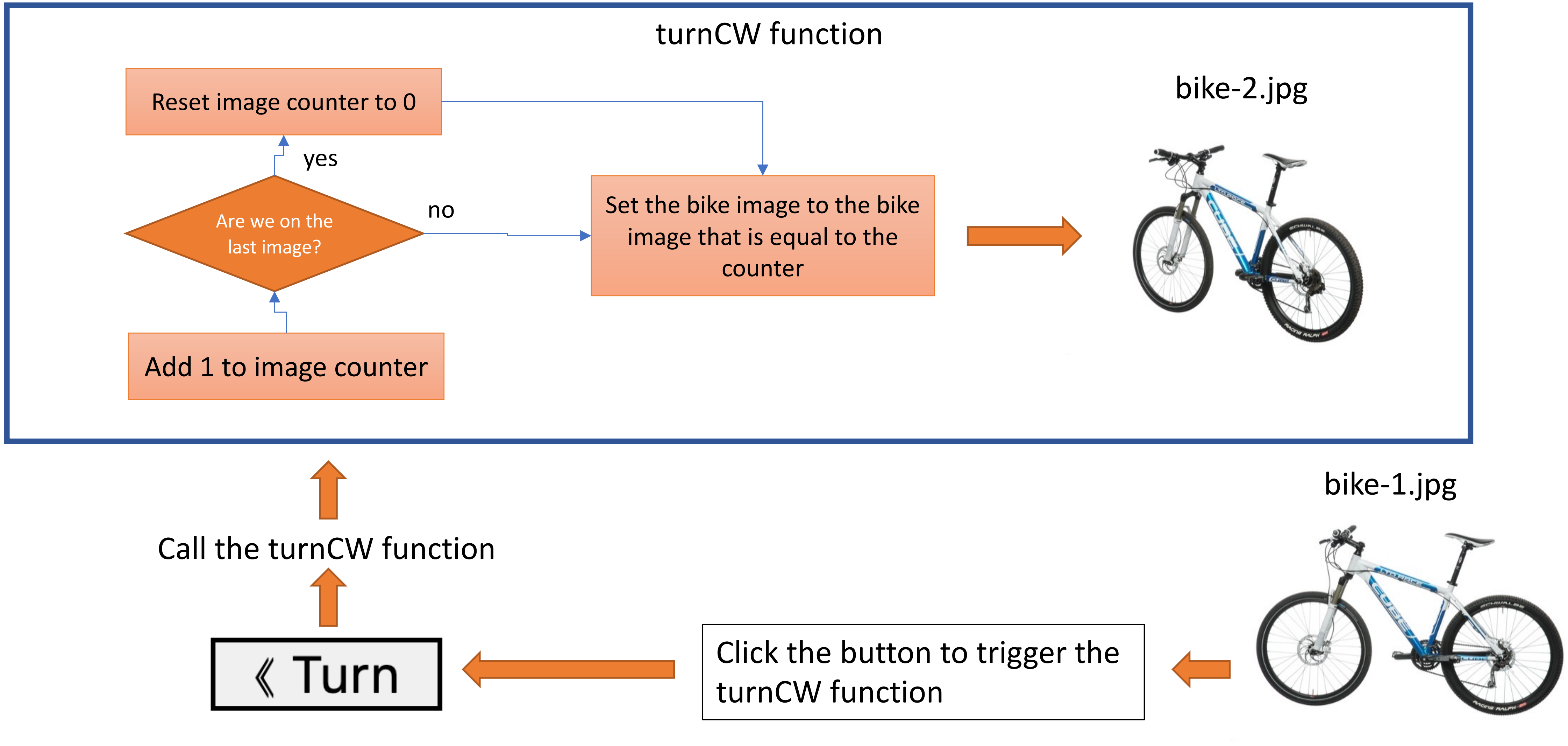
Step 6

Define our turn clockwise function

The turnCW() Function

- The turnCW function adds one to the counter variable
- The function then checks to make sure we have not reached the end of our images
  - If we have reached the end of our images we reset the counter to zero and output the first image
- Lastly the function outputs the next image (or the first image) in our array of bike images to the HTML page

turnCW() Function


turnCW function
Reset image counter to 0
yes
Are we on the last image?
no
Set the bike image to the bike image that is equal to the counter
bike-2.jpg
Add 1 to image counter
Call the turnCW function
bike-1.jpg
《 Turn
Click the button to trigger the turnCW function

turnCW() Function Code

script.js

```
function turnCW(){

   // add 1 to the counter
   counter++;

   // If the counter is equal to the length of
   // our array of bike images we have reached the
   // last slide. We then reset the counter to zero
   // to return to the first slide
   if(counter === numberOfSlides){
      counter = 0;
   }

   // Update the image on the screen
   slide.src = slides[counter].src;

} // end turnCW
```

Add the following code to the end of the script.js file

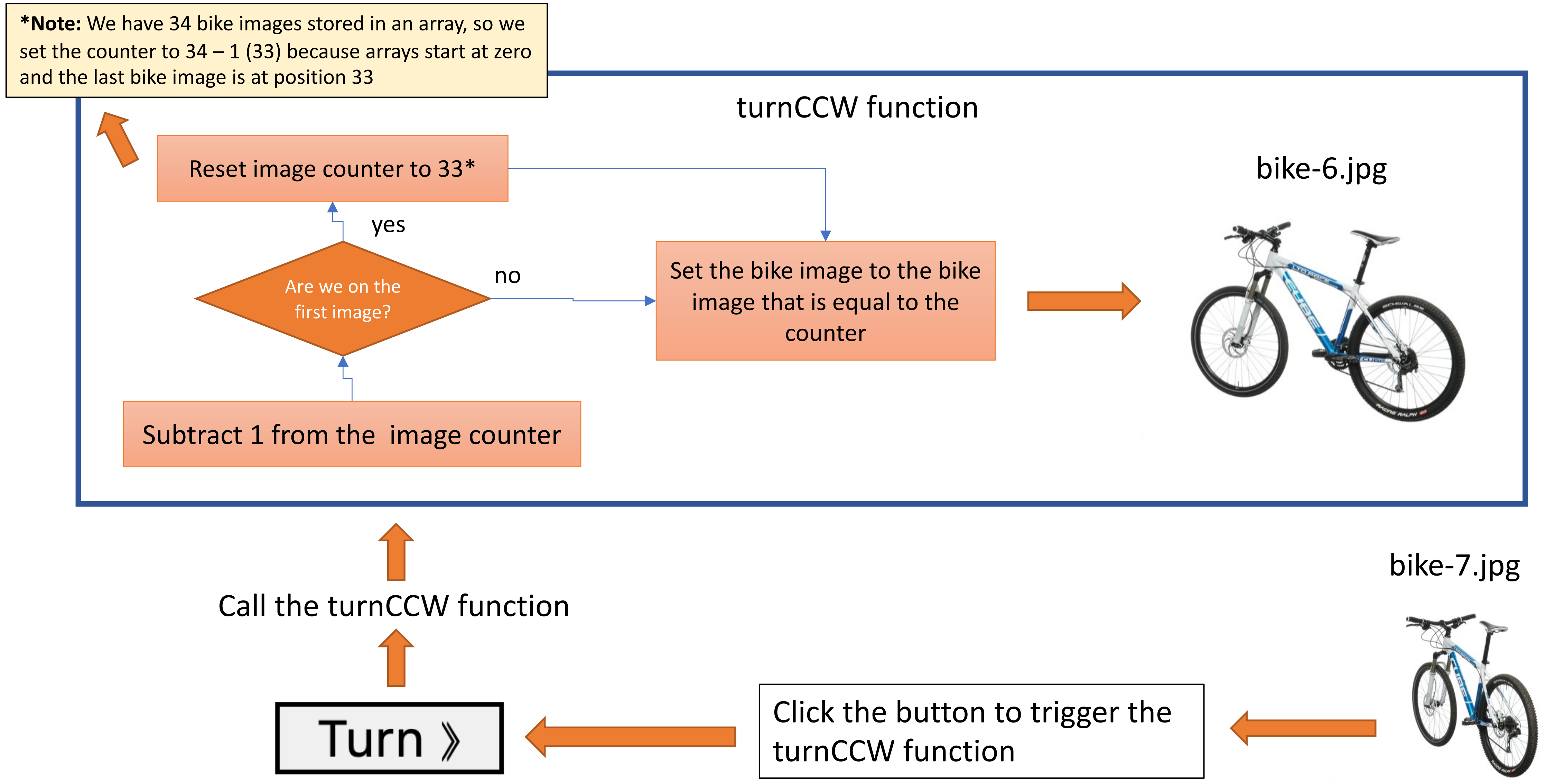
Step 7

Define our turn counter-clockwise function

The turnCCW() Function

- The turnCCW function subtracts one from the counter variable
- The function then checks to make sure we have not reached the start of our images
  - If we have reached the start of our images we reset the counter to the numberSlides variable less 1 (numberOfSlides – 1)
    - The minus 1 is because arrays are zero indexed which means to select the 34th image in the array we need to set the counter to 33 (numberOfSlides - 1)
- Lastly the function outputs the previous image (or the last image) in our array of bike images to the HTML page

turnCCW() Function



*Note: We have 34 bike images stored in an array, so we set the counter to 34 – 1 (33) because arrays start at zero and the last bike image is at position 33
turnCCW function
Reset image counter to 33*
yes
Are we on the first image?
no
Set the bike image to the bike image that is equal to the counter
Subtract 1 from the image counter
bike-6.jpg
Call the turnCCW function
Turn 》
Click the button to trigger the turnCCW function
bike-7.jpg

turnCCW() Function Code

script.js

```
function turnCCW(){

   // subtract 1 from the counter
   counter--;

   // If the counter is less than zero, we have
   // reached the beginning of our bike images.
   // We reset the counter to point to the
   // last bike image in the array
   if(counter < 0){
      counter = numberOfSlides - 1;
   }

   // Update the image on the screen
   slide.src = slides[counter].src;

} // end turnCCW
```

Add the following code to the end of the script.js file

Step 8

Add Our Click Event Handlers to the Buttons

Add Click Handlers to the Buttons

- Add a click handler to the "btnTurnCW" button
  - This event handler should call the turnCW() function
- Add a click handler to the "btnTurnCCW" button
  - This event handler should call the turnCCW() function

Button Event Handler Code

script.js

```
// Add event handlers to the buttons
btnTurnCW.addEventListener('click', turnCW);
btnTurnCCW.addEventListener('click', turnCCW);
```

Add the following code to the script.js anywhere after button variable definitions

Success!!!

(Hopefully : |)

Step 9

Testing

Testing your Application

- Your script should now be fully functional
- You should be able to click either button and turn the image clockwise or counter clockwise
- Errors
  - If your script does not function
    - Check your browser's console for errors reported by the browser

Complete

Awesome Work!!! :)