Image gallery

Now that we've looked at the fundamental building blocks of JavaScript, we'll test your knowledge of loops, functions, conditionals and events by getting you to build a fairly common item you'll see on a lot of websites — a JavaScript-powered image gallery.

Prerequisites:	Before attempting this assessment you should have already worked through all the articles in this module.
Objective:	To test comprehension of JavaScript loops, functions, conditionals, and events.

Starting point

To get this assessment started, you should go and grab the ZIP file for the example and unzip it somewhere on your computer.

> **Note**: Alternatively, you could use a site like JSBin or Thimble to do your assessment. You could paste the HTML, CSS and JavaScript into one of these online editors. If the online editor you are using doesn't have separate JavaScript/CSS panels, feel free to put them inline `<script>`/`<style>` elements inside the HTML page.

Project brief

You have been provided with some HTML, CSS and image assets and a few lines of JavaScript code; you need to write the necessary JavaScript to turn this into a working program. The HTML body looks like this:

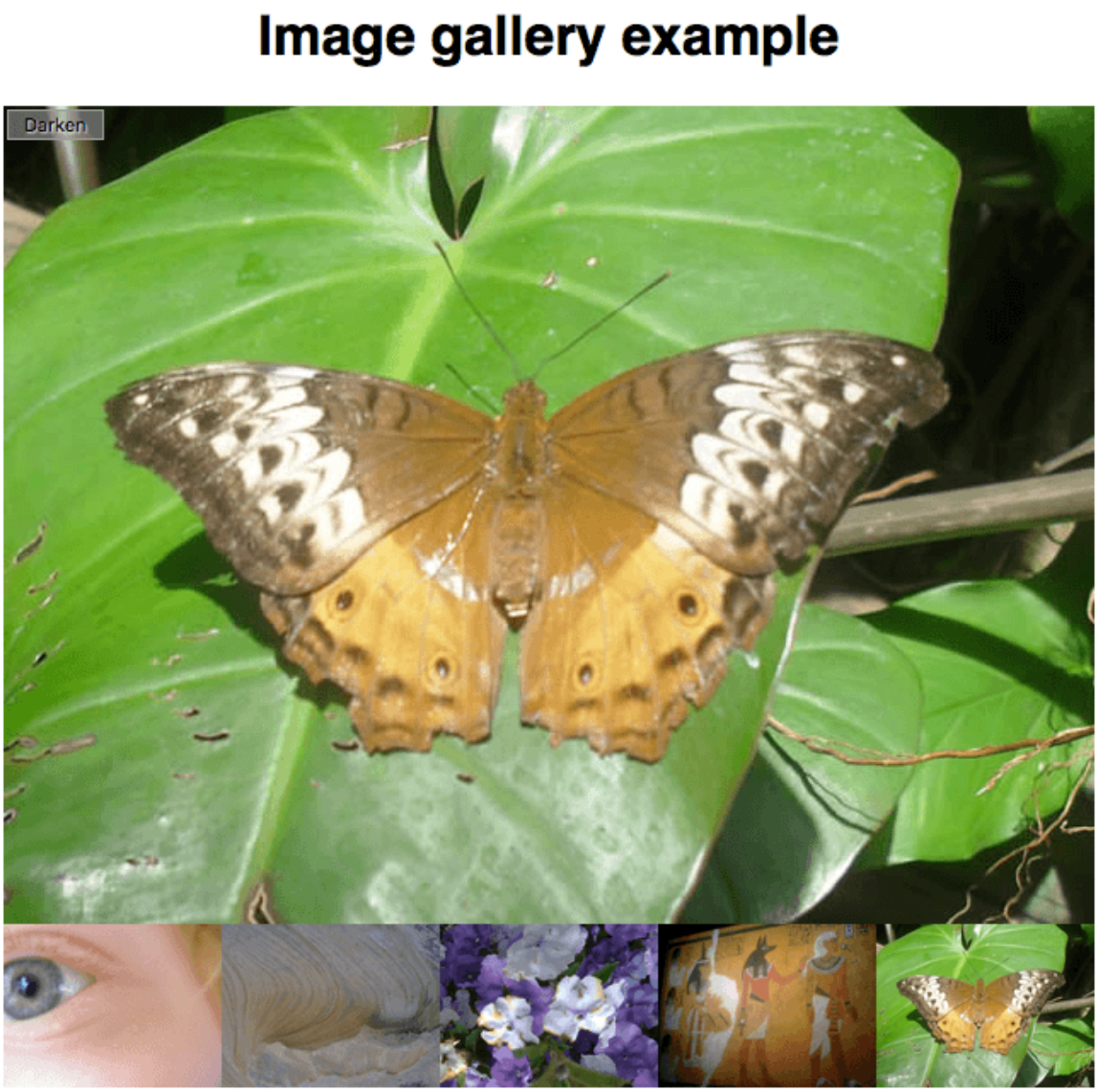
```
<h1>Image gallery example</h1>

<div class="full-img">
  <img class="displayed-img" src="images/pic1.jpg">
  <div class="overlay"></div>
  <button class="dark">Darken</button>
</div>

<div class="thumb-bar">

</div>
```

The example looks like this:



The most interesting parts of the example's CSS file:

- Absolutely position the three elements inside the `full-img` `<div>` — the `<img>` in which the full-sized image is displayed, an empty `<div>` that is sized to be the same size as the `<img>` and put right over the top of it (this is used to apply a darkening effect to the image via a semi-transparent background color), and a `<button>` that is used to control the darkening effect.

- Set the width of any images inside the `thumb-bar <div>` (so-called "thumbnail" images) to 20%, and float them to the left so they sit next to one another on a line.

Your JavaScript needs to:

- Loop through all the images, and for each one insert an `<img>` element inside the `thumb-bar <div>` that will embed that image in the page.
- Attach an `onclick` handler to each `<img>` inside the `thumb-bar <div>` so that when they are clicked, the corresponding image will be displayed in the `displayed-img <img>` element.
- Attach an `onclick` handler to the `<button>` so that when it is clicked, a darken effect is applied to the full-size image. When it is clicked again, the darken effect is removed again.

To give you more of an idea, have a look at the finished example (no peeking at the source code!)

Steps to complete

The following sections describe what you need to do.

Looping through the images

We've already provided you with lines that store a reference to the `thumb-bar <div>` inside a variable called `thumbBar`, create a new `<img>` element, set its `src` attribute to a placeholder value `xxx`, and append this new `<img>` element inside `thumbBar`.

You need to:

1. Put the section of code below the "Looping through images" comment inside a loop that loops through all 5 images — you just need to loop through five numbers, one representing each image.
2. In each loop iteration, replace the `xxx` placeholder value with a string that will equal the path to the image in each case. We are setting the value of the `src` attribute to this value in each case. Bear in mind that in each case, the image is inside the images directory and its name is `pic1.jpg`, `pic2.jpg`, etc.

Adding an onclick handler to each thumbnail image

In each loop iteration, you need to add an `onclick` handler to the current `newImage` — this should:

1. Find the value of the `src` attribute of the current image. This can be done by running the `getAttribute()` function on the `<img>` in each case, and passing it a parameter of `"src"` in each case. But how to get the image? Using `newImage` won't work, as the loop is completed before the event handlers are applied; doing it this way would result in the `src` value of the last `<img>` being returned in every case. To solve this, bear in mind that in the case of each event handler, the `<img>` is the target of the handler. How about getting the information from the event object?
2. Run a function, passing it the returned `src` value as a parameter. You can call this function whatever you like.
3. This event handler function should set the `src` attribute value of the `displayed-img` `<img>` to the `src` value passed in as a parameter. We've already provided you with a line that stores a reference to the relevant `<img>` in a variable called `displayedImg`. Note that we want a defined named function here.

Writing a handler that runs the darken/lighten button

That just leaves our darken/lighten `<button>` — we've already provided a line that stores a reference to the `<button>` in a variable called `btn`. You need to add an `onclick` handler that:

1. Checks the current class name set on the `<button>` — you can again achieve this by using `getAttribute()`.
2. If the class name is `"dark"`, changes the `<button>` class to `"light"` (using `setAttribute()`), its text content to "Lighten", and the `background-color` of the overlay `<div>` to `"rgba(0,0,0,0.5)"`.
3. If the class name not `"dark"`, changes the `<button>` class to `"dark"`, its text content back to "Darken", and the `background-color` of the overlay `<div>` to `"rgba(0,0,0,0)"`.

The following lines provide a basis for achieving the changes stipulated in points 2 and 3 above.

```
btn.setAttribute('class', xxx);
btn.textContent = xxx;
overlay.style.backgroundColor = xxx;
```

Hints and tips

- You don't need to edit the HTML or CSS in any way.

Assessment

If you are following this assessment as part of an organized course, you should be able to give your work to your teacher/mentor for marking. If you are self-learning, then you can get the marking guide fairly easily by asking on the discussion thread about this exercise, or in the #mdn IRC channel on Mozilla IRC. Try the exercise first — there is nothing to be gained by cheating!

Previous

Overview: Building blocks

In this module

Last modified: Mar 23, 2019, by MDN contributors

Related Topics

Complete beginners start here!

Complete beginners start here!

- ▸ Getting started with the Web

HTML — Structuring the Web

- ▸ Introduction to HTML
- ▸ Multimedia and embedding
- ▸ HTML tables
- ▸ HTML forms

CSS — Styling the Web

- ▸ Introduction to CSS
- ▸ Styling text
- ▸ Styling boxes
- ▸ CSS layout

JavaScript — Dynamic client-side scripting

- ▸ JavaScript first steps
- ▾ JavaScript building blocks

  JavaScript building blocks overview

  Making decisions in your code — Conditionals

  Looping code

  Functions — Reusable blocks of code

  Build your own function

  Function return values

  Introduction to events

  Assessment: Image gallery

- ▸ Introducing JavaScript objects
- ▸ Client-side web APIs

Accessibility — Make the web usable by everyone

- ▸ Accessibility guides
- ▸ Accessibility assessment

Tools and testing

- Cross browser testing

Server-side website programming

- First steps
- Django web framework (Python)
- Express Web Framework (node.js/JavaScript)

Further resources

- Advanced learning material
- Common questions

How to contribute