

# Sprint 04

## Marathon Full Stack

March 26, 2021



## Contents



<b>Engage</b> .....	<b>2</b>
<b>Investigate</b> .....	<b>3</b>
<b>Act Basic: Task 00 &gt; Secret lab</b> .....	<b>5</b>
<b>Act Basic: Task 01 &gt; Elements</b> .....	<b>10</b>
<b>Act Basic: Task 02 &gt; Image slider</b> .....	<b>14</b>
<b>Act Basic: Task 03 &gt; Weather forecast</b> .....	<b>16</b>
<b>Act Basic: Task 04 &gt; Notes with cookies</b> .....	<b>18</b>
<b>Act Advanced: Task 05 &gt; Drag'n'Drop Stones</b> .....	<b>20</b>
<b>Act Advanced: Task 06 &gt; Sort table</b> .....	<b>22</b>
<b>Act Advanced: Task 07 &gt; Marvelous list</b> .....	<b>24</b>
<b>Act Advanced: Task 08 &gt; Lazy loading</b> .....	<b>26</b>
<b>Act Advanced: Task 09 &gt; Notes with local storage</b> .....	<b>29</b>
<b>Share</b> .....	<b>31</b>

## Engage



### DESCRIPTION

You have now worked with both HTML and JavaScript. Even though you know how to include your JS scripts into HTML files, you haven't yet done any actual interaction between the two. It's time to find out how to affect the elements of your HTML web page using JS.

In this **Sprint** you will be working with the Document Object Model (DOM). It's basically a programming interface for HTML and XML documents. By using DOM, it becomes possible to connect to, modify, and control web pages using programming languages.

You will learn how DOM works, and how to use JS to manipulate your web page.

### BIG IDEA

Using DOM.

### ESSENTIAL QUESTION

How to manipulate an HTML page with JavaScript?

### CHALLENGE

Explore user interaction with a web service and interaction between two web services.



Sprint 04 | Marathon Full Stack > 2

## Investigate



### GUIDING QUESTIONS

We invite you to find answers to the following questions. By researching and answering them, you will gain the knowledge necessary to complete the challenge. To find answers, ask the students around you and search the internet. We encourage you to ask as many questions as possible. Note down your findings and discuss them with your peers.

- What are DOM and HTML DOM?
- What is the difference between `let` and `var`?
- What is the `const` keyword used for?
- Which methods are used to track events on a web page?
- What is a `querySelector`? What is it used for?
- What are `cookies`? What are they used for?
- What is `localStorage` in JS?
- What is the difference between `localStorage` and `sessionStorage`?

### GUIDING ACTIVITIES

Complete the following activities. Don't forget that you have a limited time to overcome the challenge. Use it wisely. Distribute tasks correctly.

- If you haven't understood something from `Sprint02` and/or `Sprint03` - come back to them and review.
- Read about `Browsers wars`.
- Clone your git repository issued on the challenge page in the LMS.
- Employ the full power of P2P by brainstorming with other students.
- Try to implement your thoughts in code.



## ANALYSIS

Analyze your findings. What conclusions have you made after completing guiding questions and activities? In addition to your thoughts and conclusions, here are some more analysis results.

- Be attentive to all statements of the story. Examine the given examples carefully. They may contain details that are not mentioned in the task.
- All tasks are divided into **Act Basic** and **Act Advanced**. You need to complete all basic tasks to validate the **Sprint**. But to achieve maximum points and more knowledge, consider accomplishing advanced tasks also.
- Analyze all information you have collected during the preparation stages.
- Perform only those tasks that are given in this document.
- Submit only the specified files in the required directory and nothing else. Garbage shall not pass.
- Pay attention to what is allowed. Use of forbidden stuff is considered a cheat and your challenge will be failed.
- The web page in the browser must open through `index.html`.
- The scripts must be written outside the HTML file - in a separate JS file (script.js).
- You can always use the **Console** panel to test and catch errors.
- Complete tasks according to the rules specified in the following style guides:
  - HTML and CSS: **Google HTML/CSS Style Guide**. As per section 3.1.7 **Optional Tags**, it doesn't apply. Do not omit optional tags, such as `<head>` or `<body>`
  - JavaScript:
    - **JavaScript Style Guide and Coding Conventions**
    - **JavaScript Best Practices**
- The solution will be checked and graded by students like you. **Peer-to-Peer learning**.
- Your work may also be graded by your mentor. So, be ready for that.
- Also, the challenge will pass automatic evaluation which is called **Oracle**.
- If you have any questions or don't understand something, ask other students or just Google it.

## Act Basic: Task 00



### NAME

Secret lab

### DIRECTORY

t00\_secret\_lab/

### SUBMIT

index.html, css/style.css, js/script.js

### ALLOWED

DOM

### DESCRIPTION

Create a function that changes the content and style of a web page. Use the HTML and CSS files available in the **SYNOPSIS**.

As you can see, on click of the button, the function `transformation()` is being called. Create this function inside your `script.js` file.

The function switches between two states of the web page.

State 1:

- displays **Bruce Banner** with **60px** font size and **2px** letter spacing
- has a **#ffb300** background color

State 2:

- displays **Hulk** with **130px** font size and **6px** letter spacing
- has a **#70964b** background color

It must be possible to keep endlessly switching between the two states.



Sprint 04 | Marathon Full Stack > 5



## SYNOPSIS

### HTML

```
<!DOCTYPE html>
<html lang="en">

<head>
  <meta charset="utf-8">
  <title>Secret lab</title>
  <meta name="description" content="t00. Secret lab">
  <link rel="stylesheet" href="css/style.css">
  <link href="https://fonts.googleapis.com/css2?family=Bowlby+One&display=swap"
        rel="stylesheet">
</head>

<body>
<h1>Secret lab</h1>

<p>
  Click <strong>Transform</strong>
  to transform from a man into a monster, or from a monster into a man.
</p>

<div id="lab">
  <h2 id="hero">Bruce Banner</h2>
  <input id="btn" type="button" onclick="transformation()" value="Transform">
</div>

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



## CSS

```
#lab {
  display: flex;
  align-items: center;
  justify-content: center;
  padding: 10px;
  border-radius: 8%;
  width: 500px;
  height: 500px;
  margin: auto;
  flex-direction: column;
  background: #ffb300;
  border: 6px solid #253324;
  text-align: center;
}

#btn {
  display: inline-block;
  padding: 7px 25px;
  cursor: pointer;
  box-shadow: 3px 4px 0 1px #8a2a21;
  background: #c62d1f linear-gradient(to bottom, #c62d1f 5%, #f24437 100%);
  border-radius: 18px;
  border: 3px solid #d02718;
  color: #fff;
  text-align: center;
  font: bold 25px Arial;
  text-decoration: none;
  text-shadow: 0 1px 0 #810e05;
}

#btn:hover {
  background: #f24437 linear-gradient(to bottom, #f24437 5%, #c62d1f 100%);
}

#btn:active {
  position: relative;
  top: 1px;
}

#hero {
  font: 60px 'Bowlby One', cursive;
  letter-spacing: 2px;
}
```



EXAMPLE

Default view of the page

Secret lab

Click Transform to transform Bruce Banner into a monster, or from a monster back to Bruce.

Bruce Banner

Transform

After clicking "Transform"

Secret lab

Click Transform to transform Bruce Banner into a monster, or from a monster back to Bruce.

Hulk

Transform

After clicking it again

Secret lab

Click Transform to transform Bruce Banner into a monster, or from a monster back to Bruce.

Bruce Banner

Transform

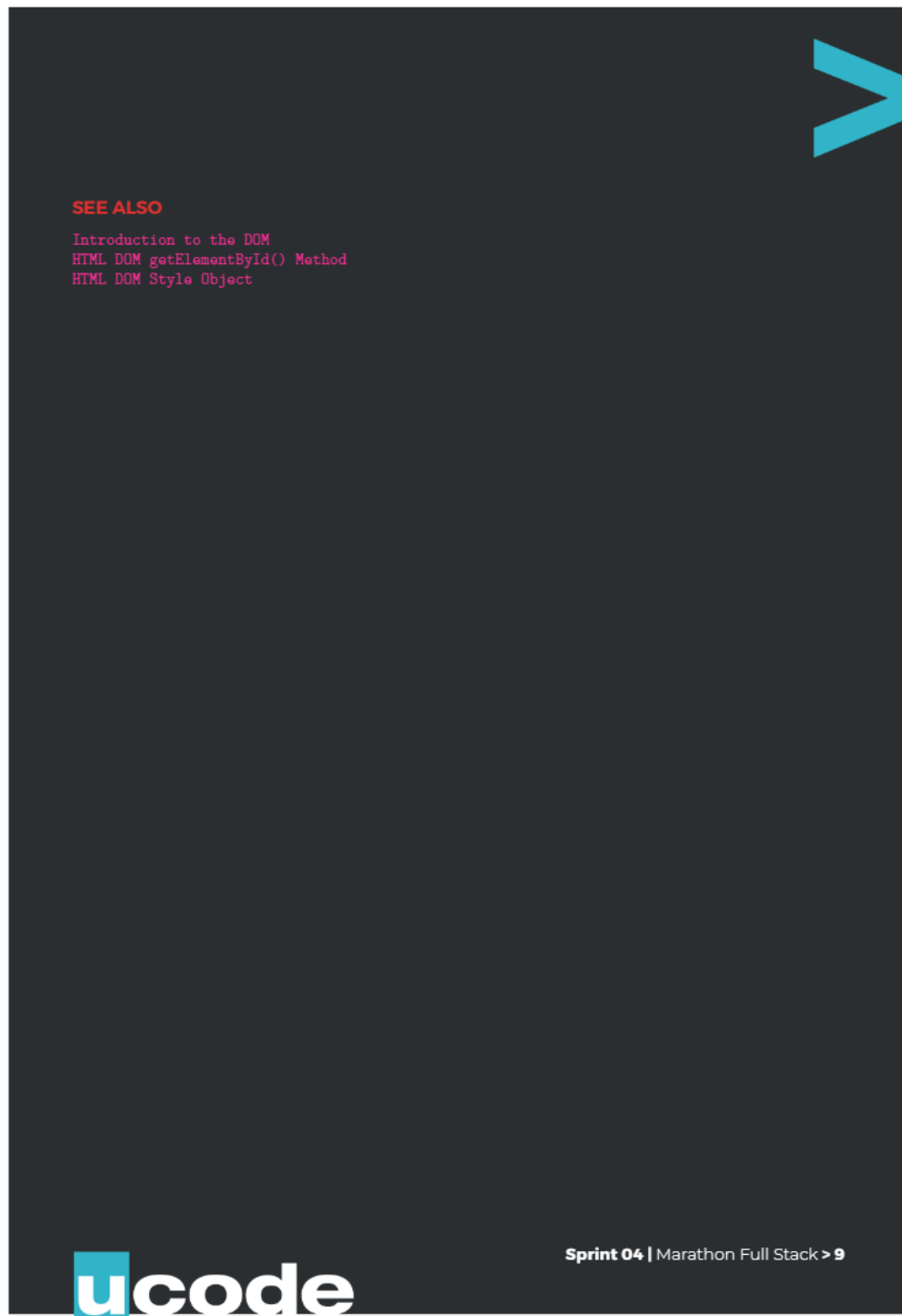
u

code

Sprint 04 | Marathon Full Stack > 8

<https://lms.ucode.world/pdf/fullstack-sprint04/sprint04>

9/32



**SEE ALSO**

```
Introduction to the DOM  
HTML DOM getElementById() Method  
HTML DOM Style Object
```

**u**code

**Sprint 04** | Marathon Full Stack > 9

## Act Basic: Task 01



### NAME

Elements

### DIRECTORY

t01\_elements/

### SUBMIT

index.html, css/style.css, js/script.js

### ALLOWED

DOM, String.\*, Array.\*

### DESCRIPTION

Create a JS script that changes the content of a web page. In this task you will need to work with HTML attributes. Use the HTML and CSS files available in the **SYNOPSIS**.

The web page contains a list of characters. Each character has some attributes. The **class** attribute specifies whether the character is good or evil. And the **data-element** attribute lists the elements that the character can control.

Compare the appearance of the web page with the screenshot in the **EXAMPLE**. Without editing the HTML and CSS files, use JS only to achieve the same result.

For each **li** element, your script must do the following:

- correct errors in attributes
  - if the **class** attribute is missing, or doesn't equal to **good**, **evil** or **unknown**, make the **class** equal to **unknown**
  - if the **data-element** attribute is missing, make the **data-element** equal to **none**
- append circle **div** elements depending on the attributes
  - for each **data-element** attribute, append a circle
  - each circle must have two class names: **elem**, and the name of the respective **data-element** attribute
  - to each **none** circle, append a **div** element with the class name **line**

Do not hardcode the solution. Your script must work, even if the contents of the list changes (items are added, removed, attributes are changed, etc.).



Sprint 04 | Marathon Full Stack > 10



## SYNOPSIS

### HTML

```
<!DOCTYPE html>
<html lang="en">

<head>
  <meta charset="utf-8">
  <title>Elements</title>
  <meta name="description" content="t01. Elements">
  <link rel="stylesheet" href="css/style.css">
  <link href="https://fonts.googleapis.com/css2?family=Bowlby+One&display=swap"
        rel="stylesheet">
</head>

<body>
  <h1>Elements</h1>
  <ul id="characters">
    <li class="good" data-element="air water earth fire">Aang</li>
    <li class="evil" data-element="fire">Zuko</li>
    <li class="good" data-element="water">Katara</li>
    <li class="good">Sokka</li>
    <li class="good" data-element="fire">Iroh</li>
    <li class="evil" data-element="fire">Azula</li>
    <li class="good" data-element="air water earth fire">Korra</li>
    <li>Suki</li>
    <li class="good" data-element="earth">Toph Beifong</li>
  </ul>
  <script src="js/script.js"></script>
</body>
</html>
```



## CSS

```
body {
  background-color: #333;
  text-align: center;
  max-width: 800px;
  margin: 0 auto;
}

h1 {color: white;}

li {
  list-style: none;
  display: inline-block;
  background-color: lightgrey;
  padding: 15px;
  margin: 5px;
  border-radius: 10%;
  text-align: center;
  border: none;
  width: 120px;
}

.good {border: 3px solid #59a440;}
.evil {border: 3px solid #ba2d29;}
.unknown {border: 3px solid #764cae;}

.elem {
  display: inline-block;
  width: 20px;
  height: 20px;
  border-radius: 50%;
  border: 2px solid white;
  margin: 3px;
}

.air {background-color: #5a8de1;}
.water {background-color: #0f1b8b;}
.earth {background-color: #496b2e;}
.fire {background-color: #9f000e;}
.none {background-color: lightgrey;}

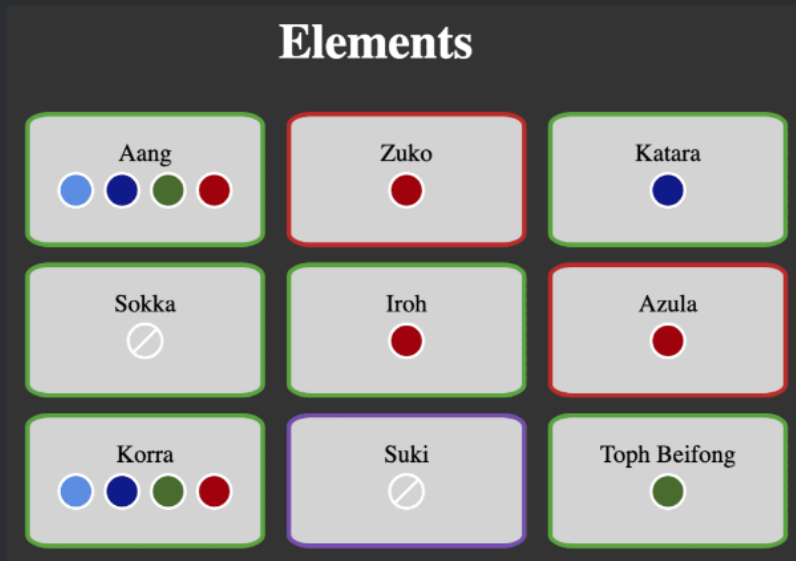
.line {
  position: relative;
  width: 20px;
  height: 2px;
  background-color: white;
  border-radius: 0;
  top: 9px;
}
```





```
transform: rotate(-225deg);  
}
```

#### EXAMPLE



#### SEE ALSO

`Element.attributes`  
[HTML | DOM appendChild\(\) Method](#)

## Act Basic: Task 02



### NAME

Image slider

### DIRECTORY

t02\_image\_slider/

### SUBMIT

index.html, css/style.css, js/script.js

### ALLOWED

DOM, String.\*, Array.\*, Object.\*, setInterval(), clearInterval()

### DESCRIPTION

Create a web page with an image slider. Follow these requirements:

- one image is shown at a time
- the "left" and "right" buttons let the user to see the previous/next image
- the images slide every 3 seconds by default until user click any button to slide the image

The design of your web page must look like the image in the **EXAMPLE**. Use CSS to achieve the result.

You can either use image files (and put them into `assets/images/*`, or use links to images online).





EXAMPLE



**u**code

Sprint 04 | Marathon Full Stack > 15



## Act Basic: Task 03



### NAME

Weather forecast

### DIRECTORY

```
t03_weather_forecast/
```

### SUBMIT

```
index.html, css/style.css, js/script.js, assets/images/*
```

### ALLOWED

DOM, Object.\*, json, Date.\*, String.\*, Class.\*, Array.\*

### DESCRIPTION

Create a weather forecast web page.

In order to get real weather information, find an API for weather forecasts. There are many to choose from.

Your forecast can be for any city/cities.

The web page must visualize weather forecast for the next several days with information about the temperature (in Celsius), and a visual representation of the weather state (whether it will be sunny, cloudy, raining, etc.).

The days of the forecast must not be hardcoded. They must depend on the current date. For example, if today is 15th of March, the forecast will display information from 15th onwards.

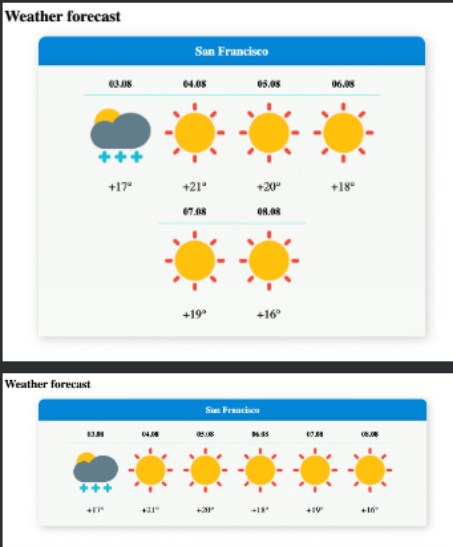
See the **EXAMPLE** images for a reference.



Sprint 04 | Marathon Full Stack > 16



EXAMPLE



SEE ALSO

[Web APIs - Introduction](#)



## Act Basic: Task 04



### NAME

Notes with cookies

### DIRECTORY

t04\_notes\_with\_cookies/

### SUBMIT

index.html, css/style.css, js/script.js

### ALLOWED FUNCTIONS

alert(), confirm(), DOM, String.\*, Array.\*, Date.\*, Object.\*, JSON.\*

### DESCRIPTION

Create a note-tracking web page that contains:

- text area field `Text input`
- button `Add to cookies`
- button `Clear cookies`
- output field `Notes archive`

Look at the **EXAMPLE**. The design is up to you.

Behavior of the web page:

- if the button `Add to cookies` is pressed - the text is added to the `Notes archive`
- if the text input area is empty - an alert box appears with a message `It's empty. Try to input something in "Text input"`.
- if the button `Clear cookies` is pressed - the confirm box appears with a message `Are you sure?`, and if the answer is positive - the output field is cleared
- `[Empty]` is displayed in the output field if the web page is launched for the first time, or the date storage has expired

In this task, you must use `cookies` for implementation. Set the expiry date to 30 days. After refreshing the page, the `Notes archive` stays the same (doesn't clear).

**EXAMPLE**

Add to cookies

Clear cookies

Text input:

Notes archive:

[Empty]

Add to cookies

Clear cookies

Text input:

Notes archive:

--> some text  
--> blablabla

**SEE ALSO**

JavaScript Cookies  
Cookies, document.cookie

## Act Advanced: Task 05



### NAME

Drag'n'Drop Stones

### DIRECTORY

```
t05_dragndrop_stones/
```

### SUBMIT

```
index.html, css/style.css, js/script.js
```

### ALLOWED

```
Object.*, Class.*, String.*, Array.*, Function.*
```

### LEGEND

With all the six stones, I could simply snap my fingers, and they would all cease to exist. I call that... mercy.



### DESCRIPTION


Create a web page with six blocks of different color, as shown in the **EXAMPLE**. Implement the following features:

- a block can have the Drag'n'Drop functionality switched on/off by clicking it
- a block with Drag'n'Drop **off** has a visible border and cannot be moved
- a block with Drag'n'Drop **on** can be moved around using the mouse

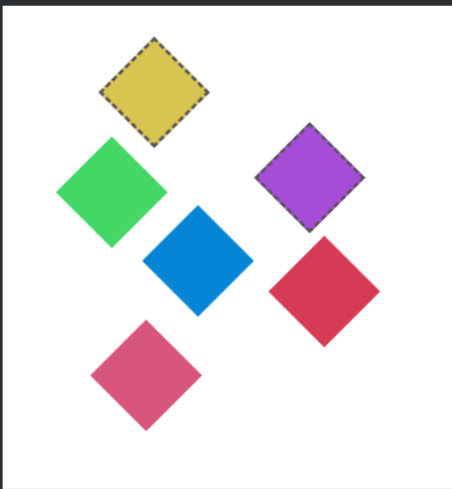
Use **events** to implement this web page.



Sprint 04 | Marathon Full Stack > 20



**EXAMPLE**



**SEE ALSO**  
[JavaScript Events](#)

**u**code

**Sprint 04** | Marathon Full Stack > 21

## Act Advanced: Task 06



### NAME

Sort table

### DIRECTORY

t06\_sort\_table/

### SUBMIT

index.html, css/style.css, js/script.js

### ALLOWED

DOM, String.\*, Array.\*, Object.\*

### DESCRIPTION

Create a web page with a table using JS. Follow these requirements:

- a table must have three columns and ten rows
- the first column must be filled with Marvel superheroes
- the second must be filled with the hero's strength
- the second must be filled with the hero's age
- the web page must have click events for the headers of the table:
  - sort the table according to the selected column by rearranging the rows. Sorting can be performed in ascending and descending order
  - show a service message `Sorting by [parameter], order: __`. Instead of `__` must be `"ASC"` or `"DESC"`

The design of your web page must look like the image in the **EXAMPLE**. Use CSS to achieve the result.

### SYNOPSIS

```
<!DOCTYPE html>
<html lang="en">

<head>
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1, maximum-scale=1" />
  <title>Sort table</title>
  <meta name="description" content="t06. Sort table">

  <link rel="stylesheet" href="css/style.css">
  <script src="js/script.js" defer></script>
</head>
```

**u**code

Sprint 04 | Marathon Full Stack > 22



```
<body>
  <main>
    <h1>Sort table</h1>

    <div id="placeholder">Placeholder with <b>HTML</b></div>
    <div id="notification">Click on cell to sort the column</div>
  </main>
</body>
</html>
```

**EXAMPLE**

Sort table Sorting by Name, order: ASC

Name	Strength	Age
Black Panther	66	53
Captain America	79	137
Captain Marvel	97	26
Hulk	80	49
Iron Man	88	48
Spider-Man	78	16
Thanos	99	1000
Thor	95	1000
Yon-Rogg	73	52



## Act Advanced: Task 07



### NAME

Marvelous list

### DIRECTORY

```
t07_marvelous_list/
```

### SUBMIT

```
index.html, css/style.css, js/script.js
```

### ALLOWED

DOM, Object.\*, String.\*, Array.\*, Set.\*

### DESCRIPTION

Create an HTML page with a list of at least three films like shown in the **EXAMPLE**. The idea is to be able to click different titles of the list and see more information on the selected one.

The page contains the following elements:

- a list of clickable titles of the films
- a block of information about a particular film from the list with the film's
  - title
  - poster image
  - the date of production
  - information
  - main actors

Add a `click` event to the document.

When the user clicks on a film's title in the list - the web page displays information about the selected title.



EXAMPLE

John Wick

Avengers: Endgame

Inception

Avengers: Endgame

April 22, 2019

Robert Downey Jr.

Chris Evans

Ian McShane

Chris Hemsworth

Scarlett Johansson

After the devastating events of Avengers: Infinity War, the universe is in ruins due to the efforts of the Mad Titan, Thanos. With the help of remaining allies, the Avengers must assemble once more in order to undo Thanos' actions and restore order to the universe once and for all, no matter what consequences may be in store.

## Act Advanced: Task 08



### NAME

Lazy loading

### DIRECTORY

```
t08_lazy_loading/
```

### SUBMIT

```
index.html, css/style.css, js/script.js
```

### ALLOWED

DOM, Object.\*, Class.\*, String.\*, Array.\*, setTimeout()

### DESCRIPTION

In this task, you'll practice lazy loading images.

Create a web page with 20 images positioned vertically, one after another.

Use the template given in the **SYNOPSIS** to add the images in your 'index.html' file. In this template

- `src="temp.jpg"` is the placeholder image (what you would see instead of the image before it loads)
- `data-src="img.jpg"` is the real image that you want on your web page (replace "img.jpg" with a link or path to your image)

When the page is loaded - unavailable images must not be displayed (display the placeholder images instead).

Implement the lazy loading effect - load the real image only when it's visible in your window.

Also, implement a message in the right top corner of the screen. The message must display the number of images that were loaded in real-time.

When all images were loaded, the message becomes green and disappears after 3 seconds.

You can either use image files (and put them into `assets/images/*`, or use links to images online).

### SYNOPSIS

```

```


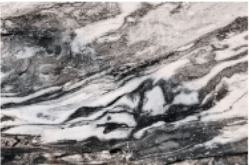



Sprint 04 | Marathon Full Stack > 26









EXAMPLE

Lazy loading

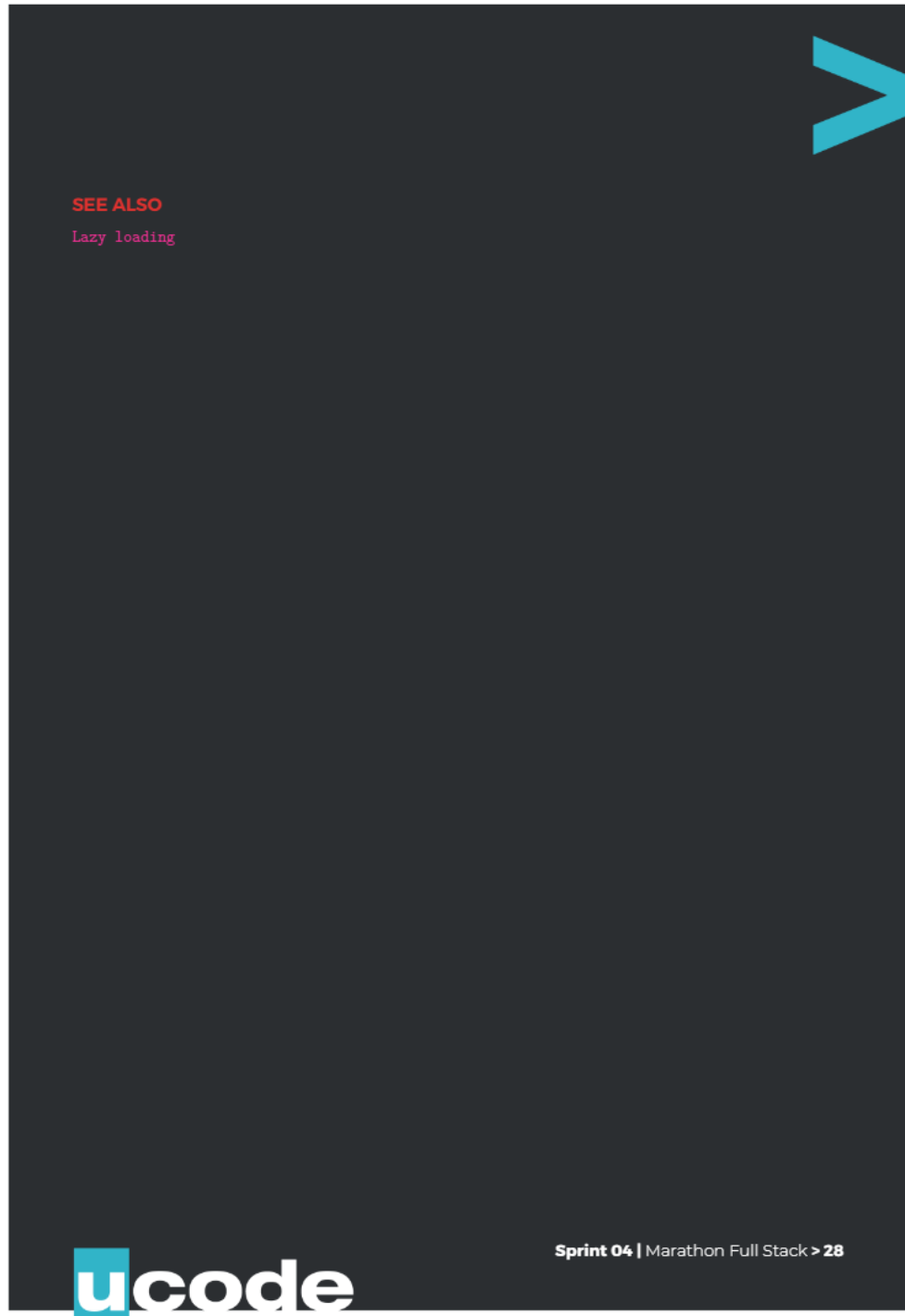


28 images loaded from 35









## Act Advanced: Task 09



### NAME

Notes with local storage

### DIRECTORY

t09\_notes\_with\_local\_storage/

### SUBMIT

index.html, css/style.css, js/script.js

### ALLOWED

alert(), confirm(), String.\*, Array.\*, Date.\*, Object.\*, RegExp.\*, DOM, localStorage.\*, JSON.\*

### DESCRIPTION

This task is very similar to the **Task 05**, but with one key difference, your web page will "remember" things using localStorage instead of cookies.

In this task, you must:

- change data storage from cookies to localStorage
- rename the buttons **Add to storage** and **Clear storage**. Of course, their behavior must be the same as the **Task 05**
- add date display as in the **EXAMPLE**



Sprint 04 | Marathon Full Stack > 29

**EXAMPLE**

Add to storage

Clear storage

Text input:

input

Notes archive:

--> some text [24.06.20, 12:03:54]  
--> blablabla [24.06.20, 12:03:58]

**SEE ALSO**[LocalStorage](#)[The complete guide to using localStorage](#)

## Share



### PUBLISHING

Last but not least, the final stage of your work is to publish it. This allows you to share your challenges, solutions, and reflections with local and global audiences. During this stage, you will discover ways of getting external evaluation and feedback on your work. As a result, you will get the most out of the challenge, and get a better understanding of both your achievements and missteps.

To share your work, you can create:

- a text post, as a summary of your reflection
- charts, infographics or other ways to visualize your information
- a video, either of your work, or a reflection video
- an audio podcast. Record a story about your experience
- a photo report with a small post

Helpful tools:

- [Canva](#) - a good way to visualize your data
- [QuickTime](#) - an easy way to capture your screen, record video or audio (macOS)
- [ScreenToGif](#) - screen, webcam, and sketchboard recorder with an integrated editor (Windows)

Examples of ways to share your experience:

- [Facebook](#) - create and share a post that will inspire your friends
- [YouTube](#) - upload an exciting video
- [GitHub](#) - share and describe your solution
- [Telegraph](#) - create a post that you can easily share on Telegram
- [Instagram](#) - share photos and stories from ucode. Don't forget to tag us :)

Share what you've learned and accomplished with your local community and the world. Use [#ucode](#) and [#CBLWorld](#) on social media.

