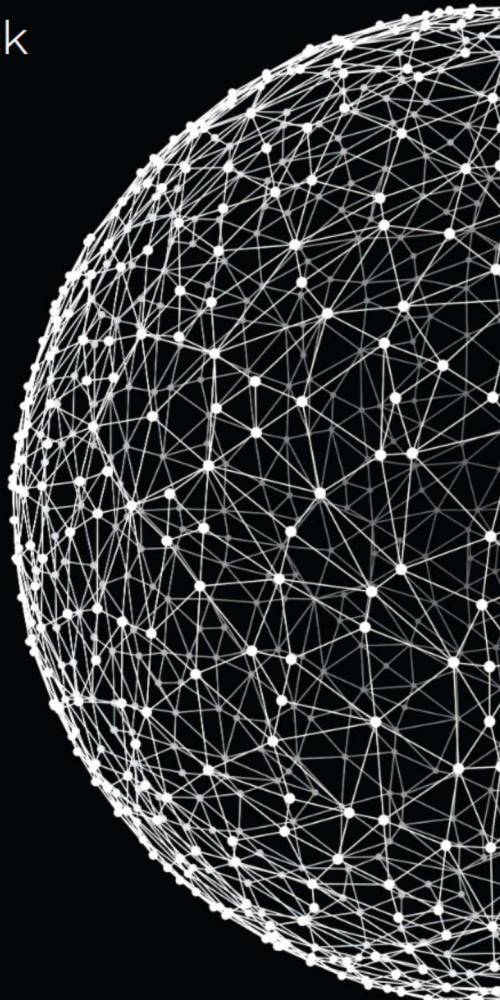


# Sprint 04

## Half Marathon Full Stack

November 15, 2021



**ucode connect**

## 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>15</b>
<b>Act Basic: Task 04 &gt; Notes with cookies</b> . . . . .	<b>17</b>
<b>Act Advanced: Task 05 &gt; Drag'n'Drop Stones</b> . . . . .	<b>19</b>
<b>Act Advanced: Task 06 &gt; Sort table</b> . . . . .	<b>21</b>
<b>Act Advanced: Task 07 &gt; Marvelous list</b> . . . . .	<b>23</b>
<b>Act Advanced: Task 08 &gt; Lazy loading</b> . . . . .	<b>25</b>
<b>Act Advanced: Task 09 &gt; Notes with local storage</b> . . . . .	<b>28</b>
<b>Share</b> . . . . .	<b>30</b>

**ucode connect**

# 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 | Half 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 solutions, 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`?
- In what ways can your JS script affect cross-browser compatibility of your web page?
- How can you test your web page for cross-browser compatibility and ensure that your script will work on different browsers?

### GUIDING ACTIVITIES

Complete the following activities. Don't forget that you have 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`.
- Learn about `common cross-browser JavaScript problems` and how to avoid them.
- 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.



Sprint 04 | Half Marathon Full Stack > 3

## 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**. This means that the complexity of the tasks increases gradually. Try to complete all tasks to get maximum points and more knowledge.
- 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.



Sprint 04 | Half Marathon Full Stack > 4

## 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.

**ucode connect**

Sprint 04 | Half 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>
```

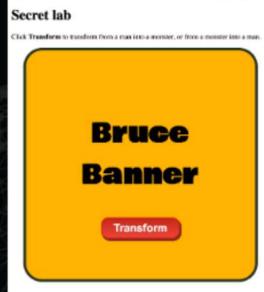
**ucode connect****Sprint 04 | Half Marathon Full Stack > 6**

## 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 'Bowby One', cursive;  
    letter-spacing: 2px;  
}
```

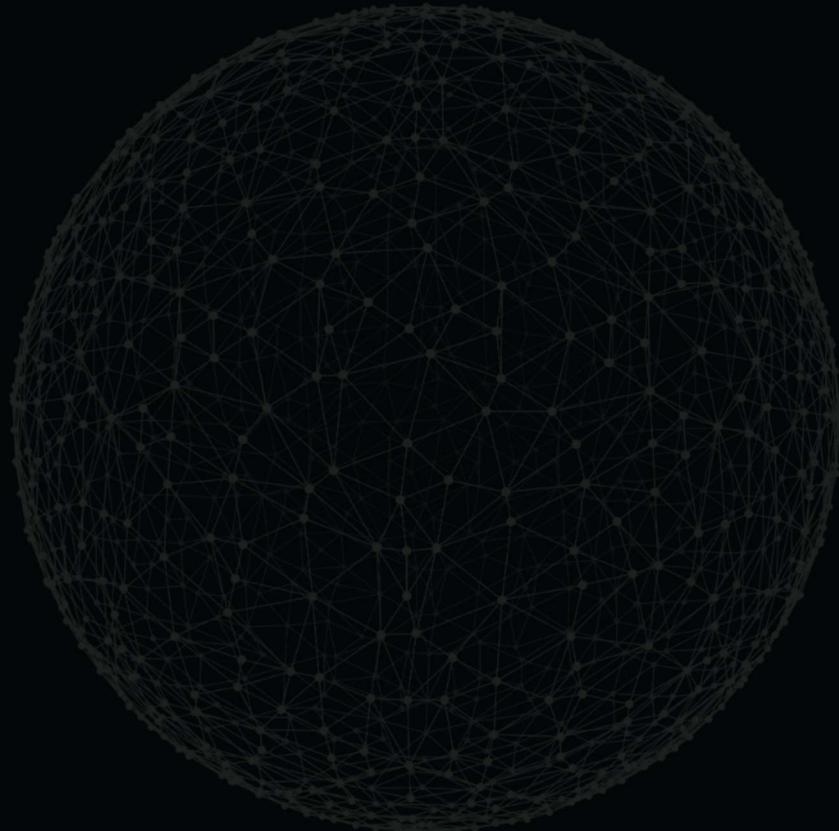


Sprint 04 | Half Marathon Full Stack &gt; 7

**EXAMPLE****Default view of the page****After clicking "Transform"****After clicking it again****ucode connect****Sprint 04 | Half Marathon Full Stack > 8**

**SEE ALSO**

[Introduction to the DOM](#)  
[HTML DOM getElementById\(\) Method](#)  
[HTML DOM Style Object](#)



**ucode connect**

Sprint 04 | Half 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 | Half 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>
```



**ucode connect**

Sprint 04 | Half Marathon Full Stack > 11

## 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;  
}
```



Sprint 04 | Half Marathon Full Stack &gt; 12

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

**EXAMPLE**

## Elements

**SEE ALSO**

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

**ucode connect****Sprint 04 | Half Marathon Full Stack > 13**

## Act Basic: Task 02

**NAME**

Image slider

**DIRECTORY**

t02\_image\_slider/

**SUBMIT**

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

**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
- there are at least 3 images (store them in assets/images/\* ..)

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

**EXAMPLE**

**ucode connect**

Sprint 04 | Half Marathon Full Stack > 14

## 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.\*

### BEFORE YOU BEGIN

In order to successfully complete this task you need to get aquainted with a few topics. Start with API. What is it? Which web services have it? Consider watching [this video](#) as an example of API usage.

Then, understand a concept of [JS promises](#).

And, in your mind, try to "connect" promises with [fetch](#). Because no serious work with API in JS could be done without it.

And of course, do not hesitate to [read MDN Web Docs](#).

### 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.

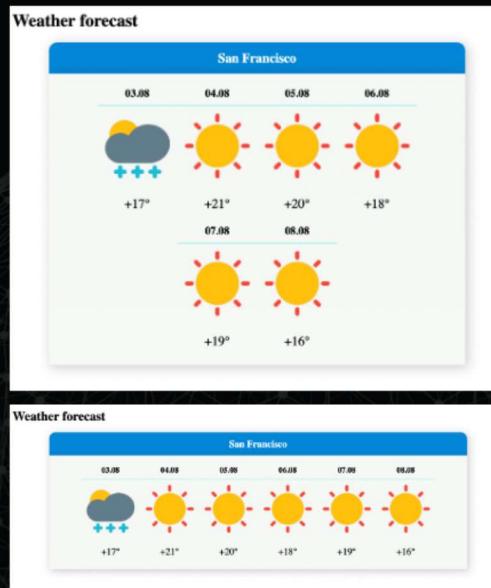
The web page must visualize weather forecast for the next several days with information about the temperature (in Celsius), and a visual representation of the weather state (whether it will be sunny, cloudy, raining, etc.).

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 | Half Marathon Full Stack > 15

**EXAMPLE****SEE ALSO**

Good summarizing article  
For those who want to understand asynchronous programming better...

**ucode connect****Sprint 04 | Half Marathon Full Stack > 16**

## 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).

**ucode connect**

Sprint 04 | Half Marathon Full Stack > 17





















