

CS50's Web Programming with Python and JavaScript_Lecture 5_JS

April 9, 2023

0.0.1 JavaScript

0.0.2 ES6 – a version of JS

JavaScript in a Web Page

```
<script>  
    alert('Hello, world!');  
</script>
```

Show 'Hello' when clicked.

```
application.py  hello0.html  events.html X
C: > Users > User > cs50WPPJ_5 > events.html > ...
1  <!DOCTYPE html>
2  <html>
3      <head>
4          <script>
5              // Function to say hello
6              function hello() {
7                  alert('Hello!');
8              }
9          </script>
10         <title>My Website</title>
11     </head>
12     <body>
13         <h1>Welcome!</h1>
14         <button onclick="hello()">Click Here!</button>
15     </body>
16 </html>
17
18
19
```

Events

- onclick
- onmouseover
- onkeydown
- onkeyup
- onload
- onblur
- ...

```
application.py  hello0.html  events.html  query.html X
C: > Users > User > cs50WPPJ_5 > query.html > ...
1  <!DOCTYPE html>
2  <html>
3      <head>
4          <script>
5              // Function to change heading to say goodbye
6              function hello() {
7                  document.querySelector('h1').innerHTML = 'Goodbye!';
8              }
9          </script>
10         <title>My Website</title>
11     </head>
12     <body>
13         <h1>Welcome!</h1>
14         <button onclick="hello()">Click Here!</button>
15     </body>
16 </html>
17
18
19
```

querySelector

- `document.querySelector('tag')`
- `document.querySelector('#id')`
- `document.querySelector('.class')`

```
application.py  hello0.html  events.html  query.html  counter0.html X
C: > Users > User > cs50WPPJ_5 > counter0.html > ...
1  <!DOCTYPE html>
2  <html>
3      <head>
4          <script>
5              let counter = 0;
6
7              function count() {
8                  counter++;
9                  document.querySelector('#counter').innerHTML = counter;
10              }
11          </script>
12         <title>My Website</title>
13     </head>
14     <body>
15         <h1 id="counter">0</h1>
16         <button onclick="count()">Click Here!</button>
17     </body>
18 </html>
19
20
```

```
application.py  hello0.html  events.html  query.html  counter0.html  counter1.html X
C: > Users > User > cs50WPPJ_5 > counter1.html > html > head
1  <!DOCTYPE html>
2  <html>
3      <head>
4          <script>
5              let counter = 0;
6
7              function count() {
8                  counter++;
9                  document.querySelector('#counter').innerHTML = counter;
10
11                  if (counter % 10 === 0) {
12                      alert('Counter is at ${counter}!');
13                  }
14              }
15          }
16      </script>
17      <title>My Website</title>
18  </head>
19  <body>
20      <h1 id="counter">0</h1>
21      <button onclick="count()">Click Here!</button>
22  </body>
23 </html>
24
25
```

0.0.3 JS in a separate file

```
y  hello0.html  events.html  query.html  counter0.html  counter1.html  counter3.html
C: > Users > User > cs50WPPJ_5 > counter3.html > ...
1  <!DOCTYPE html>
2  <html>
3      <head>
4          <script src="counter3.js"></script>
5          <title>My Website</title>
6      </head>
7      <body>
8          <h1 id="counter">0</h1>
9          <button>Click Here!</button>
10     </body>
11 </html>
12
```

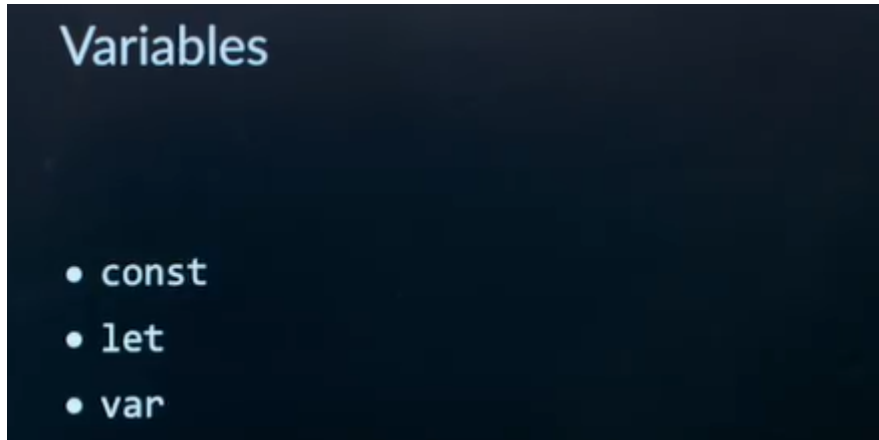
0.0.4 the .js file



```
C:\Users\User> cd C:\Users\User\cs50WPPJ_5 > .\JS counter3.js > ...
1  document.addEventListener('DOMContentLoaded', function() {
2      document.querySelector('button').onclick = count;
3  });
4
5  let counter = 0;
6
7  function count() {
8      counter++;
9      document.querySelector('#counter').innerHTML = counter;
10
11     if (counter % 10 === 0) {
12         alert('Counter is at ${counter}!');
13     }
14 }
15
```

Use of `addEventListener()` – a callback function

If there are multiple buttons, `querySelector()` will just select the first one.



let will exist in the innermost curly braces surrounding it. Outside of the curly brace where it was declared, it will not exist.

var will exist even in the outer curly braces.

```
query.html counter0.html counter1.html counter3.html JS counter3.js variables0.html X
C: > Users > User > cs50WPPJ_5 > variables0.html > ...
1  <!DOCTYPE html>
2  <html>
3      <head>
4          <script>
5              // This variable exists even outside the loop
6              if (true) {
7                  var message = 'Hello!';
8              }
9
10             alert(message);
11         </script>
12         <title>My Website</title>
13     </head>
14     <body>
15         <h1>Welcome!</h1>
16     </body>
17 </html>
```

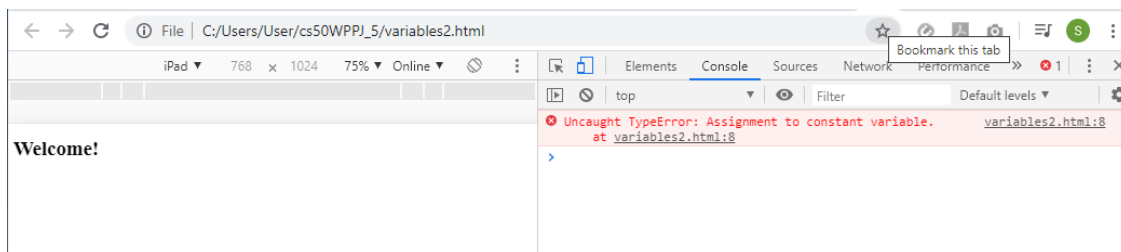
We will get an alert now.

Now, if we use **let** to define **message**, we will not see the alert.

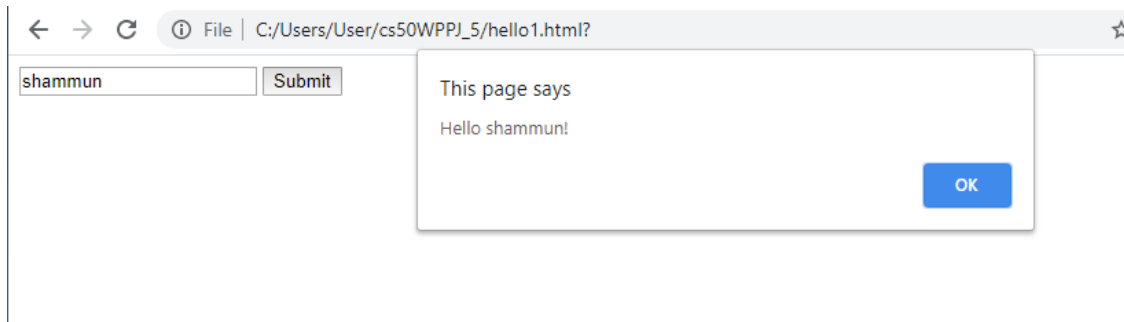
```
counter0.html counter1.html counter3.html JS counter3.js variables0.html variables1.html X
C: > Users > User > cs50WPPJ_5 > variables1.html > ...
1  <!DOCTYPE html>
2  <html>
3      <head>
4          <script>
5              // This variable does not exist outside the loop
6              if (true) {
7                  let message = 'Hello!';
8              }
9
10             alert(message);
11         </script>
12         <title>My Website</title>
13     </head>
14     <body>
15         <h1>Welcome!</h1>
16     </body>
17 </html>
```

Now, try to change **const** and it will not work.

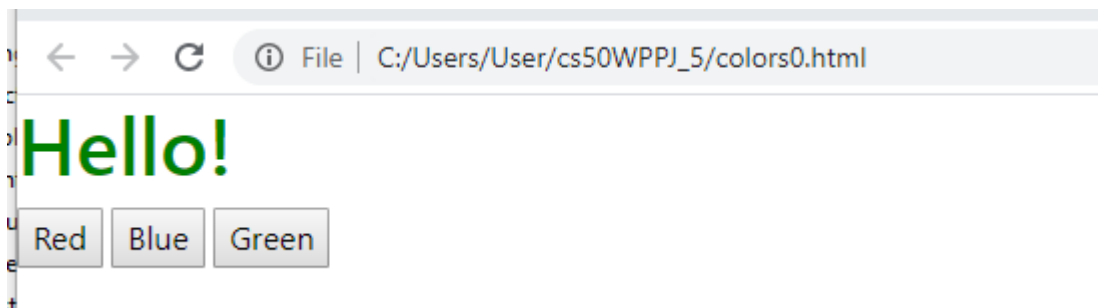
```
counter1.html  counter3.html  JS counter3.js  variables0.html  variables1.html  variables2.html X
C: > Users > User > cs50WPPJ_5 > variables2.html > ...
1  <!DOCTYPE html>
2  <html>
3      <head>
4          <script>
5              // The value of const variables cannot change
6              const message = 'Hello!';
7              message = 'Goodbye!';
8          </script>
9          <title>My Website</title>
10     </head>
11     <body>
12         <h1>Welcome!</h1>
13     </body>
14 </html>
15
16
17
18
```



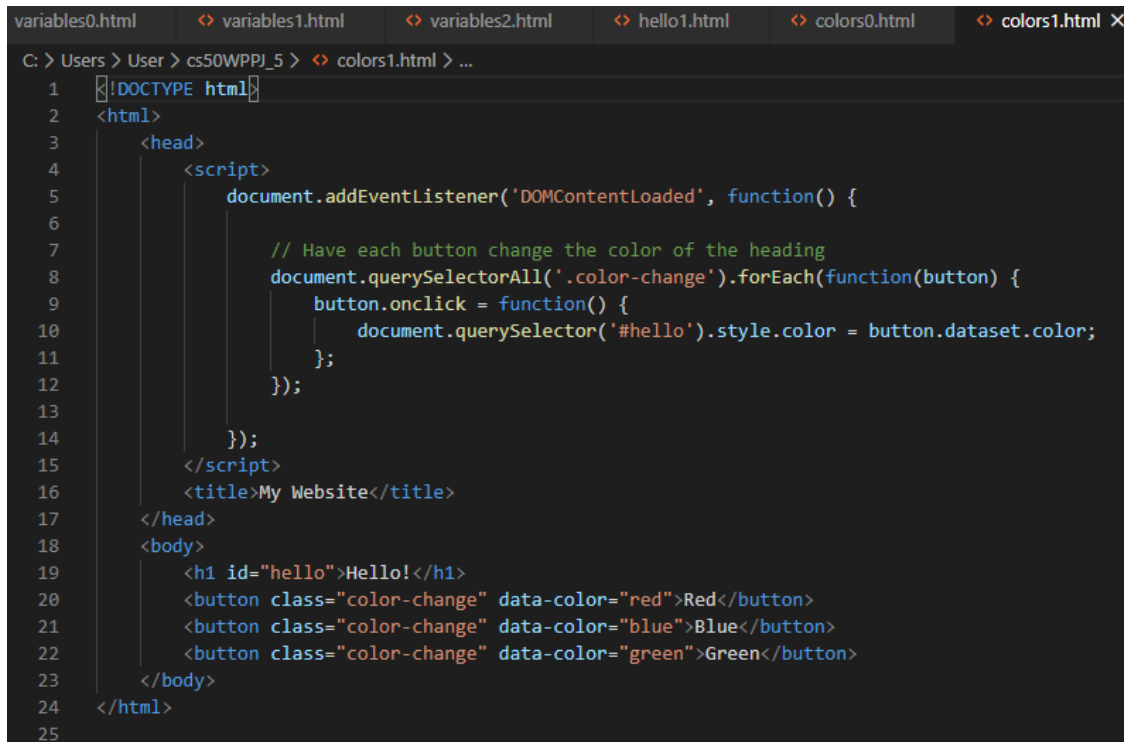
```
counter3.html  JS counter3.js  variables0.html  variables1.html  variables2.html  hello1.html X
C: > Users > User > cs50WPPJ_5 > hello1.html > ...
1  <!DOCTYPE html>
2  <html>
3      <head>
4          <script>
5              document.addEventListener('DOMContentLoaded', function() {
6                  document.querySelector('#form').onsubmit = function() {
7                      const name = document.querySelector('#name').value;
8                      alert('Hello ${name}!');
9                  };
10             });
11     </script>
12     <title>My Website</title>
13 </head>
14 <body>
15     <form id="form">
16         <input id="name" autocomplete="off" autofocus placeholder="Name" type="text">
17         <input type="submit">
18     </form>
19 </body>
20 </html>
21
```



```
C:\Users> User > cs50WPPJ_5 > colors0.html > ...
1  <!DOCTYPE html>
2  <html>
3  <head>
4  <script>
5      document.addEventListener('DOMContentLoaded', function() {
6
7          // Change font color to red
8          document.querySelector('#red').onclick = function() {
9              document.querySelector('#hello').style.color = 'red';
10             };
11
12         // Change font color to blue
13         document.querySelector('#blue').onclick = function() {
14             document.querySelector('#hello').style.color = 'blue';
15         };
16
17         // Change font color to green
18         document.querySelector('#green').onclick = function() {
19             document.querySelector('#hello').style.color = 'green';
20         };
21     });
22 </script>
23 <title>My Website</title>
24 <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/css/bootstrap.min.css" integrity="sha384-Gn5384xqQ1aoWXA+058RXPxPg6fy4"
25 </head>
26 <body>
27     <h1 id="hello">Hello!</h1>
28     <button id="red">Red</button>
29     <button id="blue">Blue</button>
30     <button id="green">Green</button>
31 </body>
32 </html>
```

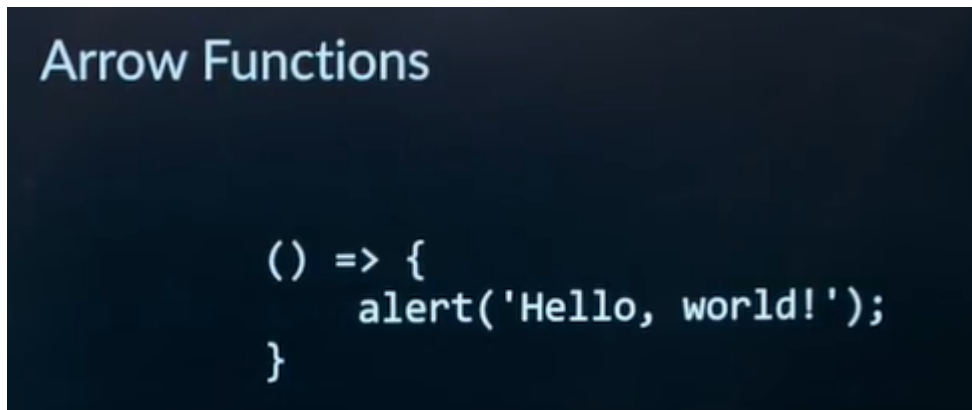


0.0.5 More efficient way



```
1  <!DOCTYPE html>
2  <html>
3    <head>
4      <script>
5        document.addEventListener('DOMContentLoaded', function() {
6
7          // Have each button change the color of the heading
8          document.querySelectorAll('.color-change').forEach(function(button) {
9            button.onclick = function() {
10              document.querySelector('#hello').style.color = button.dataset.color;
11            };
12          });
13        });
14      </script>
15      <title>My Website</title>
16    </head>
17    <body>
18      <h1 id="hello">Hello!</h1>
19      <button class="color-change" data-color="red">Red</button>
20      <button class="color-change" data-color="blue">Blue</button>
21      <button class="color-change" data-color="green">Green</button>
22    </body>
23  </html>
```

0.0.6 ==>



```
Arrow Functions

() => {
  alert('Hello, world!');
}
```

Arrow Functions

```
x => {  
    alert(x);  
}
```

Arrow Functions

```
x => x * 2
```

The following code is the same as before using ==>

```
variables1.html  variables2.html  hello1.html  colors0.html  colors1.html  colors2.html X  
C: > Users > User > cs50WPPJ_5 > colors2.html > html > body > h1#hello  
1  <!DOCTYPE html>  
2  <html>  
3      <head>  
4          <script>  
5              document.addEventListener('DOMContentLoaded', () => {  
6  
7                  // Have each button change the color of the heading  
8                  document.querySelectorAll('.color-change').forEach(button => {  
9                      button.onclick = () => {  
10                         document.querySelector('#hello').style.color = button.dataset.color;  
11                     };  
12                 });  
13             });  
14         </script>  
15         <title>My Website</title>  
16     </head>  
17     <body>  
18         <h1 id="hello">Hello!</h1>  
19         <button class="color-change" data-color="red">Red</button>  
20         <button class="color-change" data-color="blue">Blue</button>  
21         <button class="color-change" data-color="green">Green</button>  
22     </body>  
23 </html>
```

```
variables2.html  hello1.html  colors0.html  colors1.html  colors2.html  colors3.html X
C: > Users > User > cs50WPPJ_5 > colors3.html > ...
1  <!DOCTYPE html>
2  <html>
3      <head>
4          <script>
5              document.addEventListener('DOMContentLoaded', () => {
6
7                  // Change the color of the heading when dropdown changes
8                  document.querySelector('#color-change').onchange = function() {
9                      document.querySelector('#hello').style.color = this.value;
10                 };
11             });
12         </script>
13         <title>My Website</title>
14     </head>
15     <body>
16         <h1 id="hello">Hello!</h1>
17         <select id="color-change">
18             <option value="black">Black</option>
19             <option value="red">Red</option>
20             <option value="blue">Blue</option>
21             <option value="green">Green</option>
22         </select>
23     </body>
24 </html>
25
26
```



```
colors3.html  tasks0.html X  colors0.html
C: > Users > User > cs50WPPJ_5 > tasks0.html > html > head > script > document.addEventListener("DOMContentLoaded"
1  <!DOCTYPE html>
2  <html>
3    <head>
4      <script>
5        document.addEventListener('DOMContentLoaded', () => {
6
7          document.querySelector('#new-task').onsubmit = () => {
8
9            // Create new item for list
10           const li = document.createElement('li');
11           li.innerHTML = document.querySelector('#task').value;
12
13           // Add new item to task list
14           document.querySelector('#tasks').append(li);
15
16           // Clear input field
17           document.querySelector('#task').value = '';
18
19           // Stop form from submitting
20           return false;
21         };
22
23       });
24     </script>
25     <title>Tasks</title>
26   </head>
27   <body>
28     <h1>Tasks</h1>
29     <ul id="tasks">
30     </ul>
31     <form id="new-task">
32       <input id="task" autocomplete="off" autofocus placeholder="New Task" type="text">
33       <input type="submit">
34     </form>
35   </body>
36 </html>
37
```

← → ↻ ⓘ File | C:/Users/User/cs50WPPJ_5/tasks0.html

Tasks

- Attend webinar
- Create a geodashboard

Another task

Submit

Tasks

- Attend webinar
- Create a geodashboard
- Another task

The above code allows to submit blank task.

Tasks

- Attend webinar
- Create a geodashboard
- Another task
-

The following file (**tasks1.html**) prevents it.

Use **disabled** property.

```

C: > Users > User > cs50WPPJ_5 > tasks1.html > html > head > script > document.addEventListener('DOMContentLoaded') callback >
1  <!DOCTYPE html>
2  <html>
3    <head>
4      <script>
5        document.addEventListener('DOMContentLoaded', () => {
6
7          // By default, submit button is disabled
8          document.querySelector('#submit').disabled = true;
9
10         // Enable button only if there is text in the input field
11         document.querySelector('#task').onkeyup = () => {
12           document.querySelector('#submit').disabled = false;
13         };
14
15         document.querySelector('#new-task').onsubmit = () => {
16
17           // Create new item for list
18           const li = document.createElement('li');
19           li.innerHTML = document.querySelector('#task').value;
20
21           // Add new item to task list
22           document.querySelector('#tasks').append(li);
23
24           // Clear input field and disable button again
25           document.querySelector('#task').value = '';
26           document.querySelector('#submit').disabled = true;
27
28           // Stop form from submitting
29           return false;
30         };
31       };
32     </script>
33     <title>Tasks</title>
34   </head>
35   <body>
36     <h1>Tasks</h1>
37     <ul id="tasks">
38     </ul>
39     <form id="new-task">
40       <input id="task" autocomplete="off" autofocus placeholder="New Task" type="text">
41       <input id="submit" type="submit">
42     </form>
43   </body>
44 </html>

```

This code has some bugs, if we type something and then delete it, we can still submit empty task. Check if the length of the task is > 0 or not.

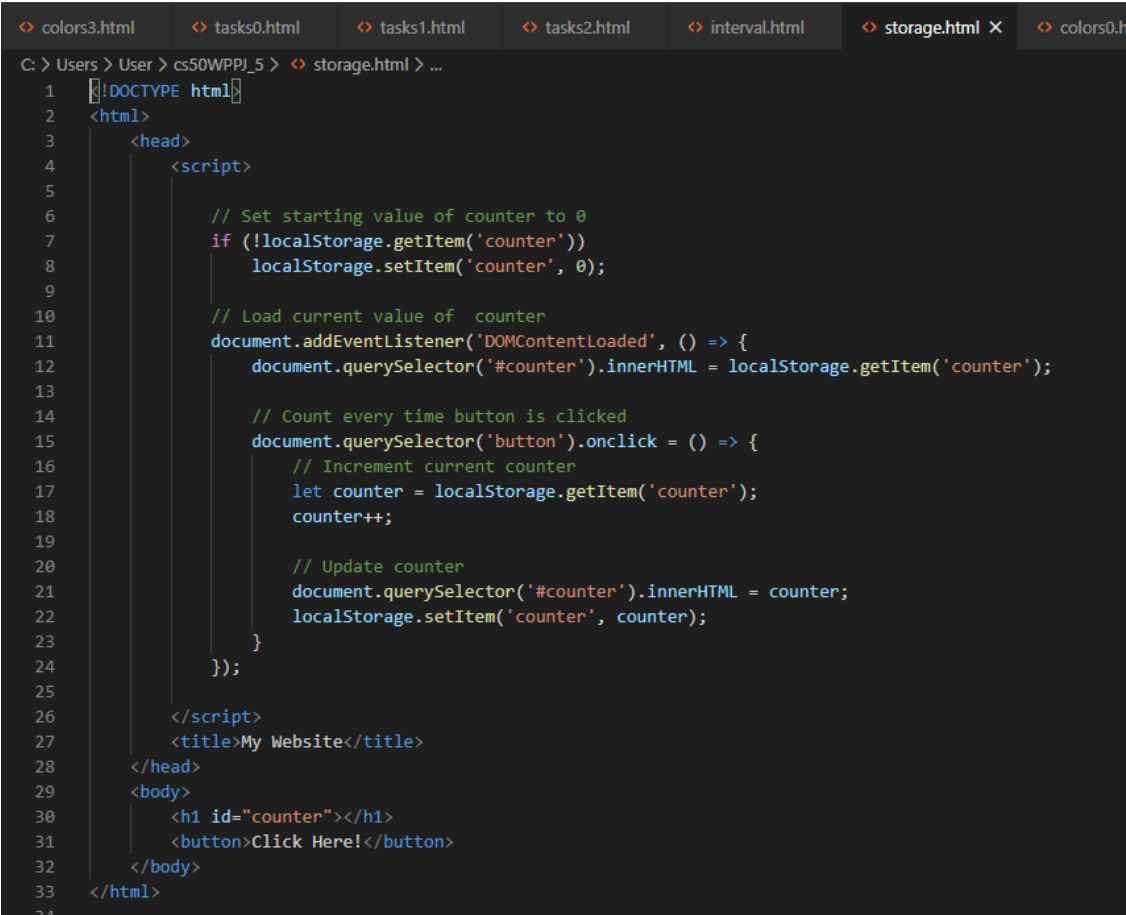
```
colors3.html tasks0.html tasks1.html tasks2.html X colors0.html
C: > Users > User > cs50WPPJ_5 > tasks2.html > ...
1  <!DOCTYPE html>
2  <html>
3      <head>
4          <script>
5              document.addEventListener('DOMContentLoaded', () => {
6
7                  // By default, submit button is disabled
8                  document.querySelector('#submit').disabled = true;
9
10                 // Enable button only if there is text in the input field
11                 document.querySelector('#task').onkeyup = () => {
12                     if (document.querySelector('#task').value.length > 0)
13                         document.querySelector('#submit').disabled = false;
14                     else
15                         document.querySelector('#submit').disabled = true;
16                 };
17
18                 document.querySelector('#new-task').onsubmit = () => {
19
20                     // Create new item for list
21                     const li = document.createElement('li');
22                     li.innerHTML = document.querySelector('#task').value;
23
24                     // Add new item to task list
25                     document.querySelector('#tasks').append(li);
26
27                     // Clear input field and disable button again
28                     document.querySelector('#task').value = '';
29                     document.querySelector('#submit').disabled = true;
30
31                     // Stop form from submitting
32                     return false;
33                 };
34
35             });
36         </script>
37         <title>Tasks</title>
38     </head>
39     <body>
40         <h1>Tasks</h1>
41         <ul id="tasks">
42         </ul>
43         <form id="new-task">
44             <input id="task" autocomplete="off" autofocus placeholder="New Task" type="text">
45             <input id="submit" type="submit">
46         </form>
47     </body>
48 </html>
```

0.0.7 Increments value from 0 in every single second and prints the value



```
1  <!DOCTYPE html>
2  <html>
3    <head>
4      <script>
5
6        document.addEventListener('DOMContentLoaded', () => {
7          setInterval(count, 1000);
8        });
9
10       let counter = 0;
11
12       function count() {
13         counter++;
14         document.querySelector('#counter').innerHTML = counter;
15       }
16
17     </script>
18     <title>My Website</title>
19   </head>
20   <body>
21     <h1 id="counter">0</h1>
22   </body>
23 </html>
```


0.0.8 Local storage – retaining inside the local machine so that even if I close the file and open it later, I still can have the value from the last time

A screenshot of a code editor with a dark theme. The editor shows a file named 'storage.html' with a JavaScript counter application. The code includes a script that initializes a counter in localStorage, updates it on page load, and increments it when a button is clicked. The HTML structure includes a head with a title 'My Website' and a body with an h1 element with id 'counter' and a button labeled 'Click Here!'. The code is as follows:

```
1  <!DOCTYPE html>
2  <html>
3    <head>
4      <script>
5
6        // Set starting value of counter to 0
7        if (!localStorage.getItem('counter'))
8          localStorage.setItem('counter', 0);
9
10       // Load current value of counter
11       document.addEventListener('DOMContentLoaded', () => {
12         document.querySelector('#counter').innerHTML = localStorage.getItem('counter');
13
14       // Count every time button is clicked
15       document.querySelector('button').onclick = () => {
16         // Increment current counter
17         let counter = localStorage.getItem('counter');
18         counter++;
19
20       // Update counter
21       document.querySelector('#counter').innerHTML = counter;
22       localStorage.setItem('counter', counter);
23     }
24   });
25
26   </script>
27   <title>My Website</title>
28 </head>
29 <body>
30   <h1 id="counter"></h1>
31   <button>Click Here!</button>
32 </body>
33 </html>
```

localStorage

0.1 AJAX

Get data even without loading page

```
tasks0.html tasks1.html tasks2.html interval.html storage.html application.py X
C: > Users > User > cs50WPPJ_5 > currency > application.py > convert
1 import requests
2
3 from flask import Flask, jsonify, render_template, request
4
5 app = Flask(__name__)
6
7
8 @app.route("/")
9 def index():
10     return render_template("index.html")
11
12
13 @app.route("/convert", methods=["POST"])
14 def convert():
15
16     # Query for currency exchange rate
17     currency = request.form.get("currency")
18     res = requests.get("https://api.fixer.io/latest", params={
19         "base": "USD", "symbols": currency})
20
21     # Make sure request succeeded
22     if res.status_code != 200:
23         return jsonify({"success": False})
24
25     # Make sure currency is in response
26     data = res.json()
27     if currency not in data["rates"]:
28         return jsonify({"success": False})
29
30     return jsonify({"success": True, "rate": data["rates"][currency]})
31
```

```
tasks0.html tasks1.html tasks2.html interval.html storage.html application.py index.html X JS index.js
C: > Users > User > cs50WPPJ_5 > currency > templates > index.html > ...
1 <!DOCTYPE html>
2 <html>
3     <head>
4         <script src="{{ url_for('static', filename='index.js') }}" type="text/javascript"></script>
5         <title>Currency Converter</title>
6     </head>
7     <body>
8         <form id="form">
9             <input id="currency" autocomplete="off" autofocus placeholder="Currency" type="text">
10             <input type="submit" value="Get Exchange Rate">
11         </form>
12         <br>
13         <div id="result"></div>
14     </body>
15 </html>
16
```

```
tasks0.html tasks1.html tasks2.html interval.html storage.html application.py index.html JS index.js X
C: > Users > User > cs50WPPJ_5 > currency > static > JS index.js > ...
1 document.addEventListener('DOMContentLoaded', () => {
2
3     document.querySelector('#form').onsubmit = () => {
4
5         // Initialize new request
6         const request = new XMLHttpRequest();
7         const currency = document.querySelector('#currency').value;
8         request.open('POST', '/convert');
9
10        // Callback function for when request completes
11        request.onload = () => {
12
13            // Extract JSON data from request
14            const data = JSON.parse(request.responseText);
15
16            // Update the result div
17            if (data.success) {
18                const contents = `1 USD is equal to ${data.rate} ${currency}.`;
19                document.querySelector('#result').innerHTML = contents;
20            }
21            else {
22                document.querySelector('#result').innerHTML = 'There was an error.';
23            }
24        }
25
26        // Add data to send with request
27        const data = new FormData();
28        data.append('currency', currency);
29
30        // Send request
31        request.send(data);
32        return false;
33    };
34
35 });
36
```

0.2 Web sockets

0.2.1 Socket.IO – a JS library

Real-time communication

0.2.2 Voting application

pip install flask_socketio

```
index.html C:\...\currency\... JS index.js C:\...\currency\... application.py C:\...\vote0 X JS index.js C:\...\vo

C: > Users > User > cs50WPPJ_5 > vote0 > application.py > ...
1 import os
2 import requests
3
4 from flask import Flask, jsonify, render_template, request
5 from flask_socketio import SocketIO, emit
6
7 app = Flask(__name__)
8 app.config["SECRET_KEY"] = os.getenv("Drmhze6EPcv0fN_81Bj-nA")
9 socketio = SocketIO(app)
10
11 @app.route("/")
12 def index():
13     return render_template("index.html")
14
15 @socketio.on("submit vote")
16 def vote(data):
17     selection = data["selection"]
18     emit("announce vote", {"selection": selection}, broadcast=True)
19
```

```
index.html C:\...\currency\... JS index.js C:\...\currency\... application.py C:\...\vote0 JS index.js C:\...\vote0\... index.html C:\...\vo

C: > Users > User > cs50WPPJ_5 > vote0 > templates > index.html > ...
1 <!DOCTYPE html>
2 <html>
3     <head>
4         <script type="text/javascript" src="//cdnjs.cloudflare.com/ajax/libs/socket.io/1.3.6/socket.io.min.js"></scrip
5         <script src="{{ url_for('static', filename='index.js') }}"></script>
6         <title>Vote</title>
7     </head>
8     <body>
9         <ul id="votes">
10         </ul>
11         <hr>
12         <button data-vote="yes">Yes</button>
13         <button data-vote="no">No</button>
14         <button data-vote="maybe">Maybe</button>
15     </body>
16 </html>
17
```

```

C:\Users\User> cd cs50WPPJ_5 > vote0 > static > JS index.js > ...
1  document.addEventListener('DOMContentLoaded', () => {
2
3      // Connect to websocket
4      var socket = io.connect(location.protocol + '//' + document.domain + ':' + location.port);
5
6      // When connected, configure buttons
7      socket.on('connect', () => {
8
9          // Each button should emit a "submit vote" event
10         document.querySelectorAll('button').forEach(button => {
11             button.onclick = () => {
12                 const selection = button.dataset.vote;
13                 socket.emit('submit vote', {'selection': selection});
14             };
15         });
16     });
17
18     // When a new vote is announced, add to the unordered list
19     socket.on('announce vote', data => {
20         const li = document.createElement('li');
21         li.innerHTML = `Vote recorded: ${data.selection}`;
22         document.querySelector('#votes').append(li);
23     });
24 });
25

```

0.2.3 Seeing vote totals and saving the vote even when we close and then reopen it

```

C:\Users\User> cd cs50WPPJ_5 > vote1 > application.py > ...
1  import os
2  import requests
3
4  from flask import Flask, jsonify, render_template, request
5  from flask_socketio import SocketIO, emit
6
7  app = Flask(__name__)
8  app.config["SECRET_KEY"] = os.getenv("SECRET_KEY")
9  socketio = SocketIO(app)
10
11  votes = {"yes": 0, "no": 0, "maybe": 0}
12
13
14  @app.route("/")
15  def index():
16      return render_template("index.html", votes=votes)
17
18
19  @socketio.on("submit vote")
20  def vote(data):
21      selection = data["selection"]
22      votes[selection] += 1
23      emit("vote totals", votes, broadcast=True)
24

```

```
ency\... application.py C:\...\vote0 JS index.js C:\...\vote0\... application.py C:\...\vote1 index.html C:\...\vote1\... JS index.js C:\...\vote1\...
C: > Users > User > cs50WPPJ_5 > vote1 > templates > index.html > ...
1 <!DOCTYPE html>
2 <html>
3   <head>
4     <script type="text/javascript" src="//cdnjs.cloudflare.com/ajax/libs/socket.io/1.3.6/socket.io.min.js"></script>
5     <script src="{{ url_for('static', filename='index.js') }}"></script>
6     <title>Vote</title>
7   </head>
8   <body>
9     <div>Yes Votes: <span id="yes">{{ votes["yes"] }}</span></div>
10    <div>No Votes: <span id="no">{{ votes["no"] }}</span></div>
11    <div>Maybe Votes: <span id="maybe">{{ votes["maybe"] }}</span></div>
12    <hr>
13    <button data-vote="yes">Yes</button>
14    <button data-vote="no">No</button>
15    <button data-vote="maybe">Maybe</button>
16  </body>
17 </html>
18
```

```
cy\... application.py C:\...\vote0 JS index.js C:\...\vote0\... application.py C:\...\vote1 index.html C:\...\vote1\... JS index.js C:\...\vote1\...
> Users > User > cs50WPPJ_5 > vote1 > static > JS index.js > ...
1 document.addEventListener('DOMContentLoaded', () => {
2
3   // Connect to websocket
4   var socket = io.connect(location.protocol + '//' + document.domain + ':' + location.port);
5
6   // When connected, configure buttons
7   socket.on('connect', () => {
8
9     // Each button should emit a "submit vote" event
10    document.querySelectorAll('button').forEach(button => {
11      button.onclick = () => {
12        const selection = button.dataset.vote;
13        socket.emit('submit vote', {'selection': selection});
14      };
15    });
16  });
17
18  // When a new vote is announced, add to the unordered list
19  socket.on('vote totals', data => {
20    document.querySelector('#yes').innerHTML = data.yes;
21    document.querySelector('#no').innerHTML = data.no;
22    document.querySelector('#maybe').innerHTML = data.maybe;
23  });
24 });
```

[]: