Element Selection (W4D2) - Learning Objectives

Element Selection

1. Given HTML that includes `<div id=”catch-me-if-you-can”>HI!</div>` , write a JavaScript statement that stores a reference to the HTMLDivElement with the id “catch-me-if-you-can” in a variable named “divOfInterest”.

```
const divOfInterest = document.getElementById('catch-me-if-you-can');
```

2. Given HTML that includes seven SPAN elements each with the class “cloudy”, write a JavaScript statement that stores a reference to a NodeList filled with references to the seven HTMLSpanElements in a variable named “cloudySpans”.

```
const cloudySpans = document.querySelectorAll('span.cloudy');
// querySelector and querySelectorAll take in CSS selectors
    // tag names are placed directly in the string: 'span'
    // classes are preceded by a .: '.cloudy'
    // ids use a #: '#catch-me-if-you-can'
// 'span.cloudy' is looking for only span tags that have the class 'cloudy'
// We could have used just '.cloudy', but if we had other tags with the same
// class they also would have been included in the collection.
```

3. Given an HTML file with HTML, HEAD, TITLE, and BODY elements, create and reference a JS file that in which the JavaScript will create and attach to the BODY element an H1 element with the id "sleeping-giant" with the content "Jell-O, Burled!".

```
window.addEventListener('DOMContentLoaded', () => {
  const message = document.createElement('h1');
  message.setAttribute('id', 'sleeping-giant');
  const text = document.createTextNode('Jell-O, Burled!')
  message.appendChild(text);
  // The following could be used instead of creating and appending a text node
  // message.innerHTML = 'Jello-O, Burled!';
  document.body.appendChild(message);
})
```

4. Given an HTML file with HTML, HEAD, TITLE, SCRIPT, and BODY elements with the SCRIPT's SRC attribute referencing an empty JS file, write a script in the JS file to create a DIV element with the id "lickable-frog" and add it as the last child to the BODY element.

```
  window.addEventListener('DOMContentLoaded', () => {
    const lastElement = document.createElement('div');
    lastElement.setAttribute('id', 'lickable-frog');
    document.body.appendChild(lastElement);
  })
```

5. Given an HTML file with HTML, HEAD, TITLE, SCRIPT, and BODY elements with no SRC attribute on the SCRIPT element, write a script in the SCRIPT block to create a UL element with no id, create an LI element with the id "dreamy-eyes", add the LI as a child to the UL element, and add the UL element as the first child of the BODY element.

```
<head>
  <script type='text/javascript'>
    const addListElement = () => {
      const listElement = document.createElement("ul");
      const listItem = document.createElement("li");
      listItem.setAttribute("id", "dreamy-eyes");
      listElement.appendChild(listItem);
      document.body.prepend(listElement);
    };
    window.onload = addListElement;
  </script>
</head>
```

6. Write JavaScript to add the CSS class "i-got-loaded" to the BODY element when the window fires the DOMContentLoaded event.

```
  window.addEventListener('DOMContentLoaded', () => {
    // *assigning* className overwrites all classes
    // document.body.className = 'i-got-loaded';

    // *adding* to classList will keep any previous classes as well
    document.body.classList.add('i-got-loaded');
  })
```

7. Given an HTML file with a UL element with the id "your-best-friend" that has six non-empty LIs as its children, write JavaScript to write the content of each LI to the console.

```
window.addEventListener("DOMContentLoaded", event => {
  const parent = document.getElementById("your-best-friend");
  const childNodes = parent.childNodes;
  for (let value of childNodes.values()) {
    console.log(value);
  }
  // Using a standard forEach, we could also look at the innerHTML of each li
  // childNodes.forEach(child => {
  //   console.log(child.innerHTML)
  // })
});
```

8. Given an HTML file with a UL element with the id "your-worst-enemy" that has no children, write JavaScript to construct a string that contains six LI tags each containing a random number and set the inner HTML property of ul#your-worst-enemy to that string.

```
// generate a random number for each list item
const getRandomInt = max => {
  return Math.floor(Math.random() * Math.floor(max));
};

// listen for DOM ready event
window.addEventListener("DOMContentLoaded", event => {
  // push 6 LI elements into an array and join
  const liArr = [];
  for (let i = 0; i < 6; i++) {
    liArr.push("<li>" + getRandomInt(10) + "</li>");
  }
  const liString = liArr.join(" ");

  // insert string into the DOM using innerHTML
  const listElement = document.getElementById("your-worst-enemy");
  listElement.innerHTML = liString;
});
```

- If we wanted to append each child as it was created instead of constructing the entire HTML we could do so:

```
const getRandomInt = max => {
  return Math.floor(Math.random() * Math.floor(max));
};

window.addEventListener("DOMContentLoaded", event => {
  const listElement = document.getElementById("your-worst-enemy");
  for (let i = 0; i < 6; i++) {
    const liItem = document.createElement('li');
    liItem.innerHTML = getRandomInt(10);
    listElement.appendChild(liItem);
  }
});
```

9. Write JavaScript to update the title of the document to the current time at a reasonable interval such that it looks like a real clock.

```
window.addEventListener("DOMContentLoaded", event => {
  const title = document.getElementById("title");
  const time = () => {
    const date = new Date();
    const seconds = date.getSeconds();
    const minutes = date.getMinutes();
    const hours = date.getHours();

    title.innerHTML = hours + ":" + minutes + ":" + seconds;
  };
  setInterval(time, 1000);
});
```