Progressoft Task - Part 2

What are Mutation Observers in JavaScript?

Most of the times we do have control over all the elements in the DOM we create in JavaScript, such as, adding, editing and removing elements, but, sometimes we don't; and that is if we're using a library that we don't have control over, or, when the DOM doesn't give us a view of what changes are happening; for example elements' width or height changes.

As an example, imagine we have an external JavaScript library that returns a random list of items at load time. And we want to know at each time the element that was returned. In this case, we only have two options:

First, setting a timeout or a time interval and print it to the console, but this way is not very efficient. The second way which is efficient, is to define a **Mutation Observer** to observe these elements and deal with them as nodes. In this way, the library returns using **querySelector** method, so, with the Mutation Observer API that provides an asynchronous way to DOM changes, we do have a solution to our case without any delay or lag.

Explain 3 Different CSS Selectors.

Let's say we do have a part of an HTML document as defined below:

```
<nav>
  <div class="header">
    <ul>
      <li class="active">Home</li>
      <li>About Us</li>
      <li>Contact</li>
    </ul>
  </div>
</nav>
<footer id="footer-main">Copyright &copy; <span>Shadi Shomali. </span> 2019</footer>
```

Three CSS Selectors we can use on the code above:

1) CSS Tag Selectors:

```
li {
	color:red;
}
```

In this case, the color of **all the list items** is red because this is a Tag Selector which access the HTML document by using tags.

2) CSS Class Selector:

```
.active {

	color:green;

}
```

Suppose in our case we want to change a list item element color to make it different than the others. In this case, we do use CSS Class Selector.

Also, Class Selectors do have a property of **accessing multiple tag elements** of HTML at the same CSS Selector.

For Example, suppose we want also to change the color of the span tag in the footer tag, the only thing we do is to add the class "active" to the <span> tag.

3) ID CSS Selector:

```
#footer-main span:hover{
	background-color: blue;
}
```

Suppose we want to change the background color of the footer element to blue on mouse hover.

The ID selector gives the HTML element **uniqueness**, in which, the selector is only valid on a single tag.

To explain the above CSS selector, it does select the <span> tag only and specifically when the <footer> tag is the parent tag and changes the background color on mouse hover.

What is the difference between SessionStorage and LocalStorage in HTML5?

SessionStorage and LocalStorage **both do extend Storage**. I could define LocalStorage as a **static storage** with a defined path that the browser renders each time at load.

For Example, we are creating a static website that has only the following:

HTML, CSS and an assets folder. At each time rendering the webpage, the CSS selectors access the path file to get the media, whether it is an image, text, styles that they are located on the server **with persistence** until explicitly deleted.

And Session Storage, I could define it **saving the changes** of a webpage in the browser tab until the tab is closed. For example, I have a Homepage that is being redirected after a login page that has a user credentials connected to the database. And I have to display the username of the user on the homepage. In this case, each user attempts to login have a different username from the other. So, I save the username in a web browser tab as a session and display it in the homepage. And in this case the SessionStorage is **non-persistent**.

Shadi Shomali