Lab 07
JavaScript Timers

Objectives:

To familiarize students with the concepts of JavaScript Animation using JavaScript Timers.

Activity Outcomes:

After this lab, the students should be able to understand the purpose and

- use of JavaScript Timers.
- Write timer functions in html for creating Website Carousel.

Instructor Note:

As pre-lab activity, read Chapter 8 from the textbook "Web Programming with HTML5, CSS, and JavaScript Pap/Psc John Dean, Year 2018. And the website https://www.w3schools.com/js/js_timing.asp for detailed functionalities.

1) Useful Concepts

A timer is a function that enables us to execute a function at a particular time.
Using timers, you can delay the execution of code so that it does not get done at the exact moment an event is triggered or the page is loaded. For example, you can use timers to change the advertisement banners on your website at regular intervals, or display a real-time clock, etc. There are two timer functions in JavaScript: setTimeout() and setInterval().

setTimeout()

The setTimeout() method calls a function or evaluates an expression after a specified number of milliseconds. The function is only executed once. If you need to repeat execution, use the setInterval() method. We use the clearTimeout() method to prevent the function from running.

Syntax: setTimeout(function, delay (in milliseconds))

This function accepts two parameters: a function, which is the function to execute, and an optional delay parameter, which is the number of milliseconds representing the amount of time to wait before executing the function (1 second = 1000 milliseconds).

clearTimeout()

The clearTimeout()method clears a timer set with the setTimeout() method. The ID value returned by setTimeout()is used as the parameter for the clearTimeout() method.

id_of_settimeout = setTimeout("javascript function", milliseconds);

If the function has not already been executed, you will be able to stop the execution by calling the clearTimeout() method.

Syntax: clearTimeout(id_of_settimeout)

setInterval()

The setInterval() function executes a function or specified piece of code repeatedly at fixed time intervals. The setInterval() method will continue calling the function until the

clearInterval() is called, or the window is closed. The ID value returned by setInterval() is used as the parameter for the clearInterval() method.

Syntax: setInterval(function, intervals(in milliseconds)).

This function also accepts two parameters: a function, which is the function to execute, and interval, which is the number of milliseconds representing the amount of time to wait before executing the function (1 second = 1000 milliseconds).

clearInterval()

The clearInterval() method clears a timer set with the setInterval () method. The ID value returned by setInterval() is used as the parameter for the clearInterval() method.

id_of_setinterval = setInterval("javascript function", milliseconds);

Then you will be able to stop the execution by calling the clearInterval() method.

clearInterval(id_of_setinterval);

2) Solved Lab Activities

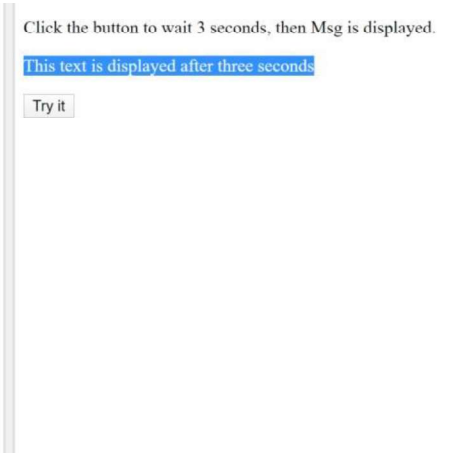
Sr. No	*Time Allocated*	*Complexity*	*CLO-Mapping*
Activity 1	*20 Minutes*	*Medium*	*CLO-1*
Activity 2	*20 Minutes*	*Medium*	*CLO-1*
Activity 3	*20 Minutes*	*Medium*	*CLO-1*

Activity 1:

In first activity of this lab, we will learn how to use setTimeOut() method to display a text message after a delay of 3 sec.

Solution:

```
<!DOCTYPE html>
<html>
<body>

<p>Click the button to wait 3 seconds, then Msg is displayed.</p>
<p id="demo"></p>

<button onclick="myFunction()">Try it</button>

<script>
var myVar;

function myFunction() {
  myVar = setTimeout(MsgFunc, 3000);
}

function MsgFunc() {
  document.getElementById("demo").innerHTML="This text is displayed after three
seconds";
}
</script>

</body>
```

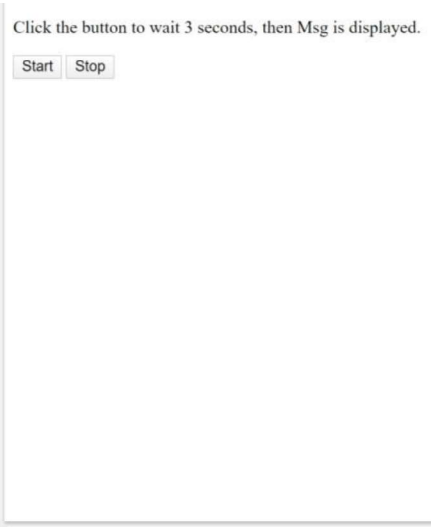


Activity 2:

In second activity of this lab, we will learn how to use clearTimeOut() method to stop the execution of a function before it is activated.

Solution:

```
<!DOCTYPE html>
<html>
<body>

<p>Click the button to wait 3 seconds, then Msg is displayed.</p>
<p id="demo"></p>

<button onclick="start()">Start</button>
<button onclick="stop()">Stop</button>

<script>
var myVar;

function start() {
  myVar = setTimeout(MsgFunc, 3000);
}


function stop() {
  clearTimeout(myVar)
}

function MsgFunc() {
  document.getElementById("demo").innerHTML="This text is displayed after three
seconds";
}
</script>
```



Activity 3:

In first activity of this lab, we will learn how to use setTimeOut() method to display a text message after a delay of 3 sec. In third activity of this lab, we will learn how to use setInterval() method for creating a clock which displays current time by updating it against each second.

Solution:

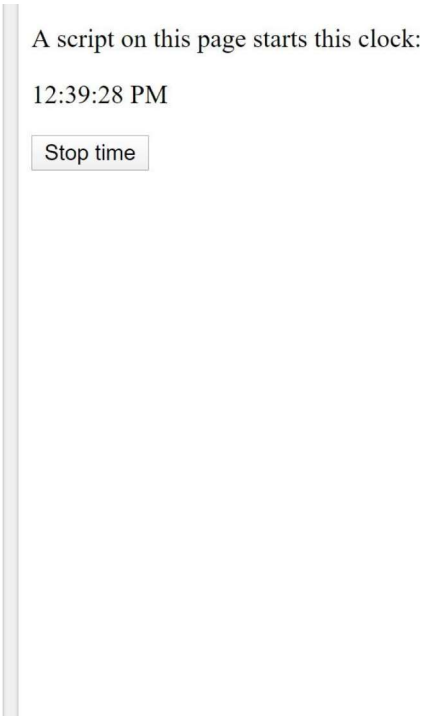
```
<!DOCTYPE html>
<html lang="en">
    <head>
        <title>Hello!</title>
        <meta charset="utf-8">
    </head>
    <body>
        <p>A script on this page starts this clock:</p>
        <p id="demo"></p>

        <script>
            var myVar = setInterval(myTimer, 1000);

            function myTimer() {
                var d = new Date();
                var t = d.toLocaleTimeString();
                document.getElementById("demo").innerHTML = t;
            }

        </script>
    </body>
</html>
```

Activity 4:

In forth activity of this lab, we will learn how to use clearInterval() method for stop the clock at any particular moment within its updating.

Solution:

```
<!DOCTYPE html>
<html>
<body>

<p>A script on this page starts this clock:</p>
<p id="demo"></p>

<button onclick="myStopFunction()">Stop time</button>

<script>
var myVar = setInterval(myTimer, 1000);

function myTimer() {
  var d = new Date();
  var t = d.toLocaleTimeString();
  document.getElementById("demo").innerHTML = t;
}

function myStopFunction() {
  clearInterval(myVar);
}
</script>

</body>
</html>
```



3) Graded Lab Tasks

Note: The instructor can design graded lab activities according to the level of difficult and complexity of the solved lab activities. The lab tasks assigned by the instructor should be evaluated in the same lab.

Lab Task 1

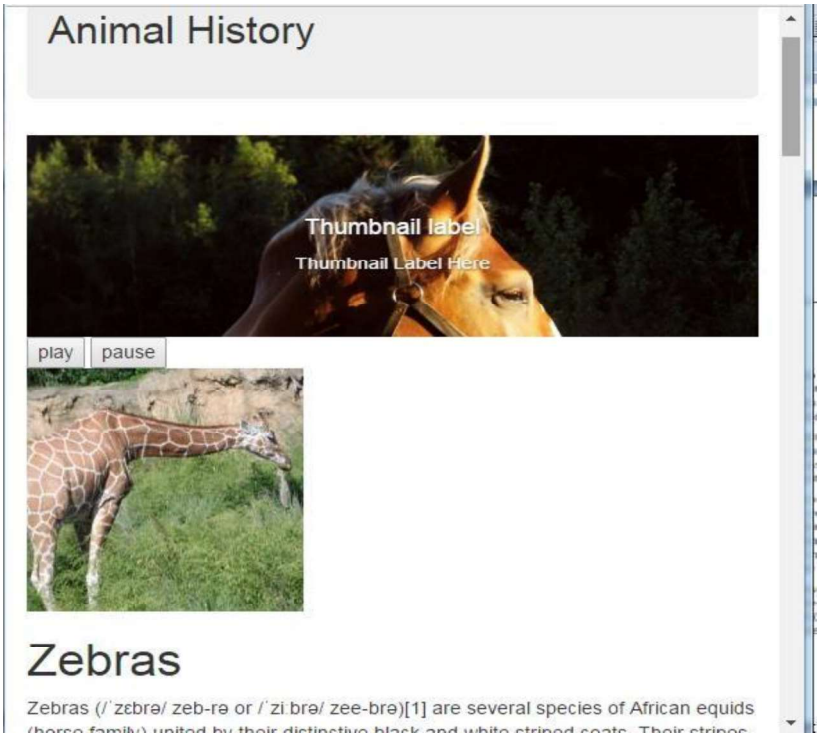
Create a Website Carousel using JavaScript for a responsive animal history web page with its layout for different screen sizes as provided below. Following are the supplementary details about different sections of the page:

1- The task should use bootstrap classes to create layout for Animal History banner, Image Slider and Animal description sections of the web page. Students can only use bootstrap.min.css as a third-party CSS library.

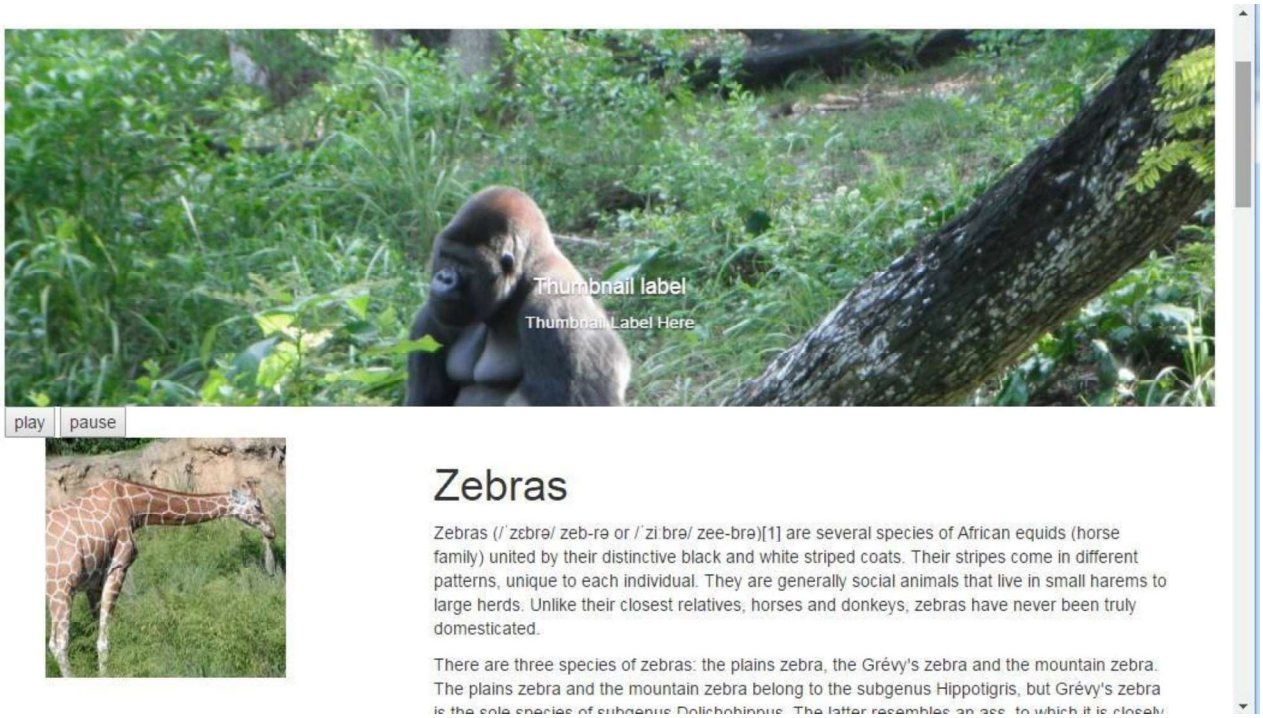
2- Students are supposed to write a JavaScript code for imageSlider with Play and Pause buttons. Slider should stop when Pause button is pressed and start when Play button is clicked.

3- It is pertinent to note that in descriptive sections of different animals, location of image and description changes with different screen sizes.

1. Extra-small, Small Screen



Medium Screen



Large Screen

Animal History

play pause
play pause

Zebras

Zebras (/ ˈzɛbrə/ zeb-rə or / ˈziːbrə/ zee-brə)[1] are several species of African equids (horse family) united by their distinctive black and white striped coats. Their stripes come in different patterns, unique to each individual. They are generally social animals that live in small harems to large herds. Unlike their closest relatives, horses and donkeys, zebras have never been truly domesticated.

There are three species of zebras: the plains zebra, the Grévy's zebra and the mountain zebra. The plains zebra and the mountain zebra belong to the subgenus Hippotigris, but Grévy's zebra is the sole species of subgenus Dolichohippus. The latter resembles an ass, to which it is closely related, while the former two are more horse-like. All three belong to the genus Equus, along with other living equids.

The unique stripes of zebras make them one of the animals most familiar to people. They occur in a variety of habitats, such as grasslands, savannas, woodlands, thorny scrublands, mountains, and coastal hills. However, various anthropogenic factors have had a severe impact on zebra populations, in particular hunting for skins and habitat destruction. Grévy's zebra and the mountain zebra are endangered. While plains zebras are much more plentiful, one subspecies, the quagga, became extinct in the late 19th century – though there is currently a plan, called the Quagga Project, that aims to breed zebras that are phenotypically similar to the quagga in a process called breeding back.

Zebras evolved among the Old World horses within the last 4 million years. It has been suggested that zebras are polyphyletic and that striped equids evolved more than once. Extensive stripes are posited to have been of little use to equids that live in low densities in deserts (like asses and some horses) or ones that live in colder climates with shaggy coats and annual shading (like some horses).[4] However, molecular evidence supports zebras as a monophyletic linage

Leopard