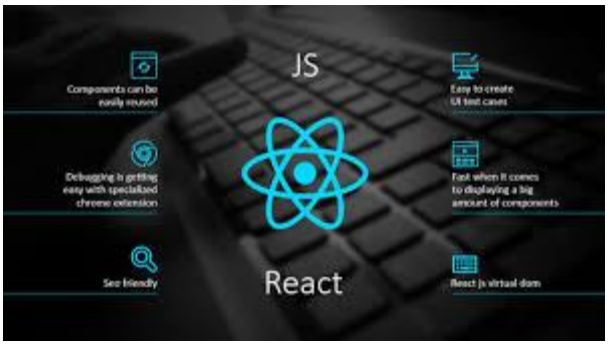
TECHNICAL CONTENT SAMPLE-HOW TO INPUT VALIDATION SET TIMEOUT IN REACTJS?

In order to have a flabbergasting effect of the animation presented in front of the viewer, it is essential to trigger the components of animation in the right manner. When you are working on a popular transition component framework like ReactJS then it is crucial to place the set Timeout() function in a precise fashion. That is due to the fact that in a complex animation there can be multiple levels where trigger is going to be required without compromising with the exact timing.

So learning how to input validation set Timeout in ReatcJS is critical if you wish to progress from being a novice to becoming a pro in the field of animation.



Introduction

The triggering of the event requires commands in terms of the setup, the duration and the intricacy in which the event appears or fades.

In this guide, we are going to learn how ReactJS set Timeout() function permits code to be executed within a set time frame after bringing some trigger into play. One such sort of trigger renders a spontaneous response when a button is pressed or a concerned page has been loaded. By the time you are done with this guide, you are going to be familiar with the process used by the professionals to trigger events subsequent to a set time via JS and how to clear timeout.

Let us go through a few custom demos down below in order to understand how things work in the real world-

set Timeout Demo 1

The function of setTimeout is going to take into consideration the following two parameters-

- The code or function to call
- The total number of milliseconds to pause prior to the execution of the function/ code

One can see an example down below where an alert box pops up 2 seconds subsequent to the click of the button.

```
<input type="button" value="click me"
  onclick="setTimeout('window.alert(\'Hello!\')', 2000)" />
```

Once you place these commands for validation, you can simply see the action trigger in no time.

set Timeout Demo 2

Please note that the following demo renders the same output as the first one, but instead move down with a function with the click of the button.The function commences the timer leading to show_alert function.

```
 <script language="Javascript">

function timeout_trigger() {
    window.alert('Hello!');
}

function timeout_init() {
    setTimeout('timeout_trigger()', 2000);
}

</script>
<input type="button" value="click me" onclick="timeout_init()" />
```

Here, it is imperative to understand that timeout has only been triggered once. In simple terms, when the timeout_triggerhas been called subsequent to 2 seconds, then it's not been called again. In order to make it called recurrently in every 2 seconds, it is suggestive to add another setTimeout() call in the end of functioning timeout_trigger.

setTimeout/ clearTimeout Demo 3

In the following example, setTimeout get back with a value storing a reference belonging to the timer. The resetting or clearing of the timer can be done with

the help of clearTimeout function. Please heed attention to the provided example down below-

```
<script language="Javascript">

var timeout;

function timeout_trigger() {
    document.getElementById('timeout_text').innerHTML = 'The timeout has been triggered';
}

function timeout_clear() {
    clearTimeout(timeout);
    document.getElementById('timeout_text').innerHTML = 'The timeout has been cleared';
}

function timeout_init() {
    timeout = setTimeout('timeout_trigger()', 3000);
    document.getElementById('timeout_text').innerHTML = 'The timeout has been started';
}

</script>

<div>
    <input type="button" value="test timeout" onclick="timeout_init()" />
    <input type="button" value="clear timeout" onclick="timeout_clear()" />
</div>
<div id="timeout_text"></div>
```

At the point where timeout_init() is applied, the concerned timeout reference is basically stored in “timeout” variable. The name that you allot to the variable is not going to make any difference, but it is crucial that it has a universalscope. That is one of the key reasons why we have put “var timeout,” affirmation at the commencement of code.

A click on “test timeout” kick-starts the timer and at the end one can clear the timeout simply by clicking “clear timeout” button.

setTimeout Application to create a countdown demo 4

For you last but not the least, we are leaving no stone unturned as the demo unveils a countdown in which numbers count down right from 10 to 0 displaying a new update each second. The step is basically accomplished via

decrementing of the counter number, then subsequently calling setTimeoutat the ultimate phase of function call timeout.

The following is the demo in the form of example code-

```
<script language="Javascript">

var countdown;
var countdown_number;

function countdown_init() {
    countdown_number = 11;
    countdown_trigger();
}

function countdown_trigger() {
    if(countdown_number > 0) {
        countdown_number--;
        document.getElementById('countdown_text').innerHTML = countdown_number;
        if(countdown_number > 0) {
            countdown = setTimeout('countdown_trigger()', 1000);
        }
    }
}

function countdown_clear() {
    clearTimeout(countdown);
}

</script>
```

Conclusion

As through various demos, we tried to help you understand various insights of ReactJSsetTimeout() function. We learned how it can leads to the occurrence of the desired effect on a webpage after a set time of a trigged command like the click of a button. For instance, you wish to disable the form button once it has been clicked in order to prevent a double form submission, but re-enable the same again in the fraction of a few passing seconds. We also understood how we can cancel the callbackfrom happeningvia clearTimeout() function, in case some other action has been done indicating the fact that the timeout is not a prerequisite anymore.

The results that we can procure out of the commands mentioned above are checked for their legitimacy at the professional level. That is the reason why you can apply them whenever you come across the timeout commands the next time and fetch best results in no time.