Full Stack Development

Week 11:

Aim: Write a Program on jQuery Traversing and filtering

Description:

In jQuery, traversing means moving through or over the HTML elements to find, filter or select a particular or entire element.
Based on the traversing purposes following methods are Categorized as follows:

Tree Traversing:
Ancestors:

- **parent()**
  it gives parent element of specified selector
  Syntax:
  $(selector).parent();

- **parents()**
  it gives all ancestor elements of the specified selector.
  Syntax:
  $(selector).parents();

- **parentsUntil()**
  it gives all ancestor elements between specified selector and arguments.
  Syntax:
- $(selector).parentsUntil(selector, filter element)
- $(selector).parentsUntil(element, filter element)

- **offsetParent()**
  it gives the first positioned parent element of specified selector.
  Syntax:
  $(selector).offsetParent();

- **closest()**
  it gives the first ancestor of the specified selector.
  Syntax:
  $(selector).closest(selector);

  $(selector).closest(selector, context);

$(selector).closest(selection);

$(selector).closest(element);

Descendants:

- **children()**
  it gives the children of each selected elements, optionally filtered by a selector.
  Syntax:
- $(selector).children();

- **find()**
  it gives descendant elements of specified elements, filtered by a selector, jQuery object, or element.
  Syntax:
  $(selector).find('selector to find');

Siblings:

- **siblings()**
  it gives all siblings of the specified selector.
  Syntax:
  $(selector).siblings();

- **next()**
  it gives the next sibling element of the specified selector.
  Syntax:
- $(selector).next();

- **nextAll()**
  it gives all next sibling elements of the specified selector.
  Syntax:
- $(selector).nextAll();

- **nextUntil()**
  it gives all next sibling elements between specified selector and arguments.
  Syntax:
- $(selector).nextUntil();

- **prev()**
  it gives the previous sibling element of the specified selector.
  Syntax:
- $(selector).prev(selector);

- $(selector).prev()

- **prevAll()**
  it gives all previous sibling elements of the specified selector.
  Syntax:
- $(selector).prevAll(selector, filter element)

- $(selector).prevAll(element, filter element)

- **prevUntil()**
  it gives all previous sibling elements between specified selector and arguments.
  Syntax:
- $(selector).prevUntil(selector, filter element)
- $(selector).prevUntil(element, filter element)

Filtering

- **first()**
  it gives the first element of the specified selector.
  Syntax:
- $(selector).first();

- **last()**
  it gives the last element of the specified selector.
  Syntax:
- $(selector).last();

- **eq()**
  it gives an element with a specific index number of the specified selector.
  Syntax:
- $(selector).eq(index);
- $(selector).eq(indexFromEnd);

- **filter()**
  it remove/detect an elements that are matched with specified selector.
  Syntax:
  $(selector).filter(selector)

  $(selector).filter(function)

  $(selector).filter(selection)

  $(selector).filter(elements)

- **has()**
  it gives all elements that have one or more elements within, that are matched with specified selector.
  Syntax:
- $(selector).has(selector);

- **is()**
  it checks if one of the specified selector is matched with arguments.
  Syntax:
  .is(selector)

  .is(function)

.is(selection)

.is(elements)

- **map()**
  Pass each element in the current matched set through a function, producing a new jQuery object containing the return values
  Syntax:
  .map(callback)

- **slice()**
  it selects a subset of specified selector based on its argument index or by start and stop value.
  Syntax:
- $(selector).slice(start, end);
- $(selector).slice(start);

Miscellaneous Traversing

- **add()**
  it add all elements to set of matched elements to manipulate them at the same time.
  Syntax:
  $(selector).add(selector to add);

- **addBack()**
  it add the previous set of elements on the stack to the current set, optionally filtered by a selector.
  Syntax:
  $(selector).addBack();

- **andSelf()**
  Deprecated 1.8 which is alias of addBack().
  Syntax:
  $(selector).addSelf();

- **contents()**
  it gives all direct children, including text and comment nodes, of the specified selector.
  Syntax:
  $(selector).contents();

- **not()**
  it gives all elements that do not match with specified selector.
  Syntax:
  $(selector).not(selector);

- **end()**
  it is most recent filtering operation in the current chain and return the set of

matched elements to its previous state and it doesn't accept any arguments.
Syntax:

- $(selector).each(callback function);

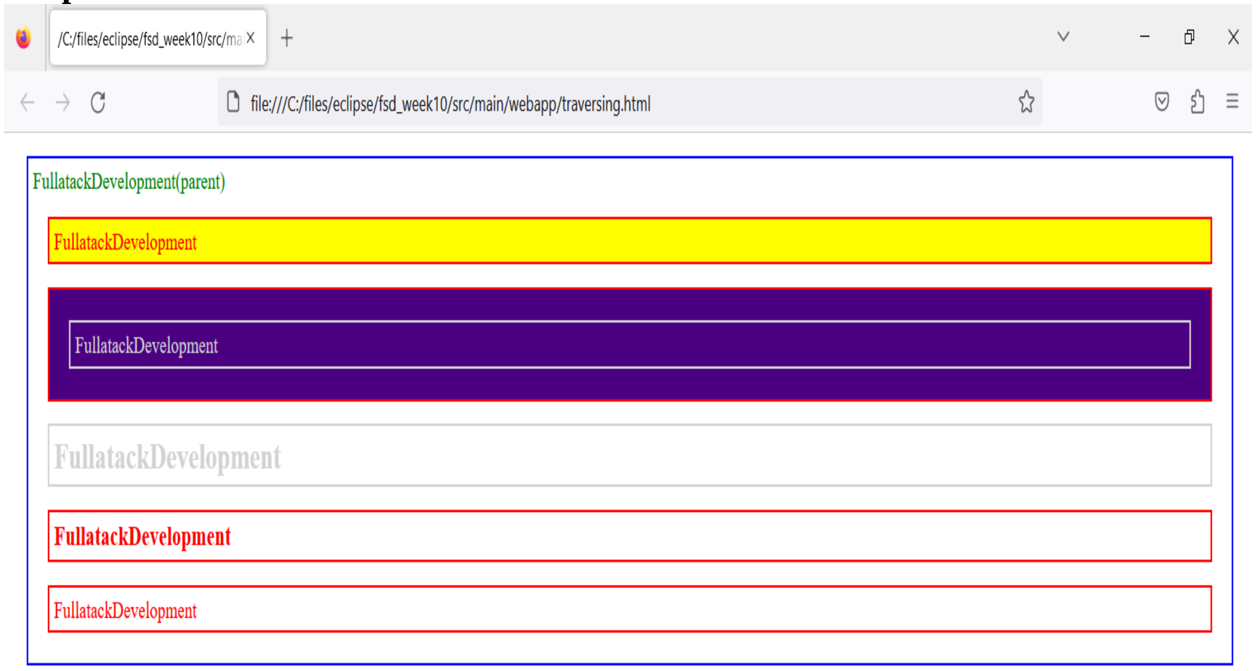
Collection Manipulation

- **each()**
  it iterates over the DOM elements and execute call back function

Example 1:

```
<!DOCTYPE html>
<html>

<head>
<style>
.siblings * {
        display: block;
        border: 2px solid lightgrey;
        color: lightgrey;
        padding: 5px;
        margin: 15px;
}
</style>
<script src="jquery.min.js">
</script>
<script>
                $(document).ready(function() {
                        $("h2").siblings().css({
                                "color": "red",
                                "border": "2px solid red"
                        });
                        $("h2").parent().css({
                                "color": "green",
                                "border": "2px solid blue"
                        });
                        $("p").first().css(
                        "background-color", "yellow");
                        $("p").has("span").css(
                        "background-color", "indigo");

                });
        </script>
</head>

<body class="siblings">

        <div>
                FullatackDevelopment(parent)
                <p>FullatackDevelopment</p>
                <p>
                        <span>FullatackDevelopment</span>
                </p>
                <h2>FullatackDevelopment</h2>
                <h3>FullatackDevelopment</h3>
                <p>FullatackDevelopment</p>
        </div>

</body>

</html>
```

Output:



/C:/files/eclipse/fsd_week10/src/ma

file:///C:/files/eclipse/fsd_week10/src/main/webapp/traversing.html

```
FullatackDevelopment(parent)
FullatackDevelopment
FullatackDevelopment
FullatackDevelopment
FullatackDevelopment
FullatackDevelopment
```

11a)

jQuery filter()

The **filter()** method returns the elements matching the specified criteria. If elements do not match the criteria, they are removed from the selection.

It can take a ***selector*** or a ***function*** as its arguments for filtering the set of matched elements. When using ***selector***, the method filters the elements don't match the given selector. If we use ***function***, the method filters the elements that don't match the given function. Generally, this method is used to reduce the search for an element in the set of selected elements.

Syntax

Using *selector*

1. $(selector).filter(selector)

Using *function*

$(selector).filter(function(index))

The parameter values of this function are defined as follows.

selector: It is an optional attribute. It could be a JQuery object or a selector expression. We can also use the comma-separated list of expressions to apply multiple filters at once. It can be written as follows:

1. filter("id1, #id2")

function: It is also an optional parameter. This parameter specifies a function that runs for every element in the group. The element is kept if the function returns true. Otherwise, on returning false, the element is removed.

The ***index*** argument represents the position of the element in the set. It begins with the **0** position.

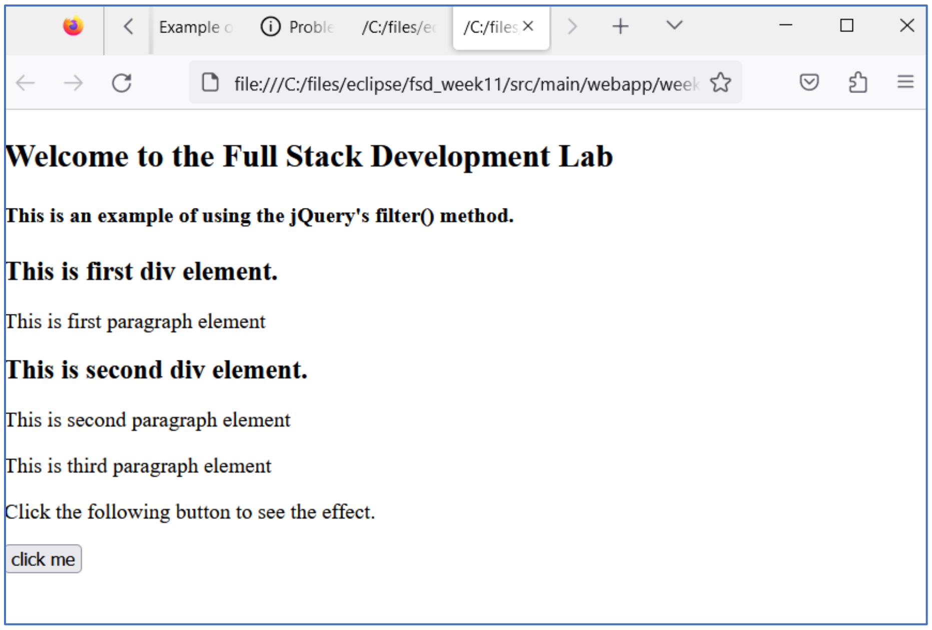
Let's see some examples to understand how to use the **filter()** method.

we are using the **selector** attribute of the **filter()** function. Here, the **filter()** function returns all paragraph elements with class name ***para***. There are some div elements, paragraph elements and others. There are three paragraph elements out of four related to the class ***para***.

We have to click the given button to see the result.

```
<!DOCTYPE html>
<html>
<head>
<style>
div {
        font-size: 20px;
        font-weight: bold;
}
</style>
<script src="jquery.min.js"></script>
<script>
        function fun(){
        $(document).ready(function(){
          $("p").filter(".para").css({"background": "yellow"});
        });
        }
        </script>
</head>

<body>
        <h2>Welcome to the Full Stack Development Lab</h2>
        <h4>This is an example of using the jQuery's filter() method.</h4>
        <div id="div1">This is first div element.</div>

        <p class="para">This is first paragraph element</p>
        <div id="div2">This is second div element.</div>
        <p class="para">This is second paragraph element</p>
        <p class="para">This is third paragraph element</p>
        <p>Click the following button to see the effect.</p>
        <button onclick="fun()">click me</button>
</body>
</html>
```

Output



Welcome to the Full Stack Development Lab

This is an example of using the jQuery's filter() method.

This is first div element.

This is first paragraph element

This is second div element.

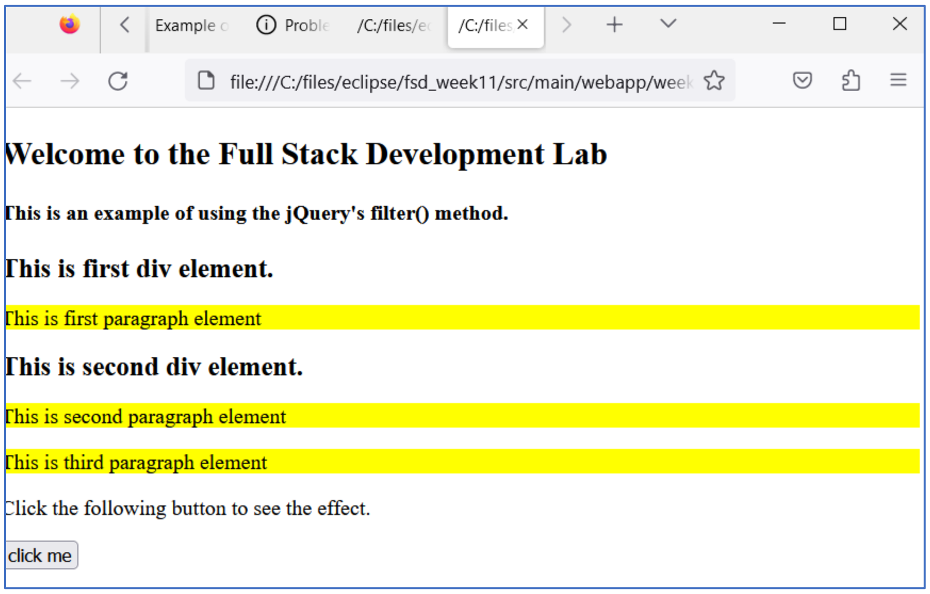
This is second paragraph element

This is third paragraph element

Click the following button to see the effect.

click me

After clicking the button, we can see that the function returns the paragraph elements related to the class name **para**.



Welcome to the Full Stack Development Lab

This is an example of using the jQuery's filter() method.

This is first div element.

This is first paragraph element

This is second div element.

This is second paragraph element

This is third paragraph element

Click the following button to see the effect.

click me

Result: Thus, in the above programs successfully executed without errors Using jQuery traversing and filtering in eclipse editor**.**