UNIT-4

jQuery

S Lakshmisri
Asst Prof.
Department of CSE
RGUKT-AP.

INTRODUCTION

- jQuery is a small, light-weight and fast JavaScript library. It is cross-platform and supports different types of browsers.
- jQuery is a JavaScript Library.
- jQuery greatly simplifies JavaScript programming.
- It is also referred as ?write less do more? because it takes a lot of common tasks that requires many lines of JavaScript code to accomplish, and binds them into methods that can be called with a single line of code whenever needed.
- The purpose of jQuery is to make it much easier to use JavaScript on your website.

jQuery Features

- HTML manipulation
- DOM manipulation
- DOM element selection
- CSS manipulation
- Effects and Animations

Why jQuery is required

- It is very fast and extensible.
- It facilitates the users to write UI related function codes in minimum possible lines.
- It improves the performance of an application.
- Browser's compatible web applications can be developed.
- It uses mostly new features of new browsers.

Many of the biggest companies on the web use jQuery. Some of these companies are:

- Microsoft
- Google
- IBM
- Netflix

Adding jQuery to Your Web Pages

There are several ways to start using jQuery on your web site. You can:

- Download the jQuery library from jQuery.com
- Include jQuery from a CDN, like Google

Downloading jQuery

There are two versions of jQuery available for downloading:

- Production version - this is for your live website because it has been minified and compressed
- Development version - this is for testing and development (uncompressed and readable code)

Both versions can be downloaded from jQuery.com.

The jQuery library is a single JavaScript file, and you reference it with the HTML <script> tag (notice that the <script> tag should be inside the <head> section):

```
<head>
<script src="jquery-3.5.1.min.js"></script>
</head>
```

jQuery CDN

- If you don't want to download and host jQuery yourself, you can include it from a CDN (Content Delivery Network).

Google is an example of someone who host jQuery:

jQuery Syntax

- The jQuery syntax is tailor-made for **selecting** HTML elements and performing some **action** on the element(s).

  Basic syntax is: **$(*selector*).*action*()**
- A $ sign to define/access jQuery
- A (*selector*) to "query (or find)" HTML elements
- A jQuery *action*() to be performed on the element(s)

Examples:

- $(this).hide() - hides the current element.
- $("p").hide() - hides all <p> elements.
- $(".test").hide() - hides all elements with class="test".
- $("#test").hide() - hides the element with id="test".

The Document Ready Event

You might have noticed that all jQuery methods in our examples, are inside a document ready event:

```
$(document).ready(function()
{
// jQuery methods go here...

});
```

- This is to prevent any jQuery code from running before the document is finished loading (is ready).
- It is good practice to wait for the document to be fully loaded and ready before working with it. This also allows you to have your JavaScript code before the body of your document, in the head section.

Here are some examples of actions that can fail if methods are run before the document is fully loaded:

- Trying to hide an element that is not created yet
- Trying to get the size of an image that is not loaded yet

jQuery Selectors

jQuery selectors are one of the most important parts of the jQuery library.

- jQuery selectors allow you to select and manipulate HTML element(s).
- jQuery selectors are used to "find" (or select) HTML elements based on their name, id, classes, types, attributes, values of attributes and much more.
- All selectors in jQuery start with the dollar sign and parentheses: $().

The element Selector

- The jQuery element selector selects elements based on the element name.
- You can select all <p> elements on a page like this:

  $("p")

Example

- When a user clicks on a button, all <p> elements will be hidden:

```
<!DOCTYPE html>
<html>
<head>
<script

src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>
```

```
$(document).ready(function(){
  $("button").click(function(){
    $("p").hide();
  });
});
</script>
</head>
<body>

<h2>This is a heading</h2>

<p>This is a paragraph.</p>
<p>This is another paragraph.</p>

<button>Click me to hide paragraphs</button>

</body>
</html>
```

The #id Selector

- The jQuery *#id* selector uses the id attribute of an HTML tag to find the specific element.
- An id should be unique within a page, so you should use the #id selector when you want to find a single, unique element.
- To find an element with a specific id, write a hash character, followed by the id of the HTML element:

  $("#test")

Example

- When a user clicks on a button, the element with id="test" will be hidden:

```
<!DOCTYPE html>
<html>
<head>
<script

src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.

js"></script>
```

```
<script>
$(document).ready(function(){
  $("button").click(function(){
    $("#test").hide();
  });
});
</script>
</head>
<body>

<h2>This is a heading</h2>

<p>This is a paragraph.</p>
<p id="test">This is another paragraph.</p>

<button>Click me</button>

</body>
</html>
```

The .class Selector

- The jQuery *.class* selector finds elements with a specific class.
- To find elements with a specific class, write a period character, followed by the name of the class:

  $(".test")

Example

- When a user clicks on a button, the elements with class="test" will be hidden:

```
<!DOCTYPE html>
<html>
<head>
<script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>
<script>
$(document).ready(function(){
  $("button").click(function(){
    $(".test").hide();
  });
});
</script>
</head>
```

```
<body>

<h2 class="test">This is a heading</h2>

<p class="test">This is a paragraph.</p>
<p>This is another paragraph.</p>

<button>Click me</button>

</body>
</html>
```

More Examples of jQuery Selectors

1. $("*")-- Selects all elements

```
<!DOCTYPE html>
<html>
<head>
<script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>
<script>
$(document).ready(function(){
  $("button").click(function(){
    $("*").hide();
  });
});
</script>
</head>
<body>
```

```
<h2>This is a heading</h2>

<p>This is a paragraph.</p>
<p>This is another paragraph.</p>

<button>Click me</button>

</body>
</html>
```

<u>2. $("[href]")--Selects all elements with an href attribute</u>

```
<!DOCTYPE html>
<html>
<head>
<script

src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.

js"></script>
```

```
<script>
$(document).ready(function(){
  $("button").click(function(){
    $("[href]").hide();
  });
});
</script>
</head>
<body>

<h2>This is a heading</h2>

<p>This is a paragraph.</p>
<p>This is another paragraph.</p>
<p><a href="https://releases.jquery.com/">Jquery cdn link</a></p>
<p><a href="https://jquery.com/download/">Jquery download</a></p>
<button>Click me</button>

</body>
</html>
```

jQuery - Attributes

- Some of the most basic components we can manipulate when it comes to DOM elements are the properties and attributes assigned to those elements.
- Most of these attributes are available through JavaScript as DOM node properties. Some of the more common properties are
- className
- tagName
- id
- href
- title
- rel
- src

Consider the following HTML markup for an image element –

<img id = "imageid" src = "image.gif" alt = "Image" class = "myclass" title = "This is an image"/>

SYNTAX

Return the value of an attribute:

```
    $(selector).attr(attribute)
```

Set the attribute and value:

```
    $(selector).attr(attribute,value)
```

Set attribute and value using a function:

```
    $(selector).attr(attribute,function
                  (Index,currentvalue))
```

Set multiple attributes and values:

```
    $
(selector).attr({attribute:value, attribute:val
ue,...})
```

- In this element's markup, the tag name is img, and the markup for id, src, alt, class, and title represents the element's attributes, each of which consists of a name and a value.
- jQuery gives us the means to easily manipulate an element's attributes and gives us access to the element so that we can also change its properties

Get Attribute Value

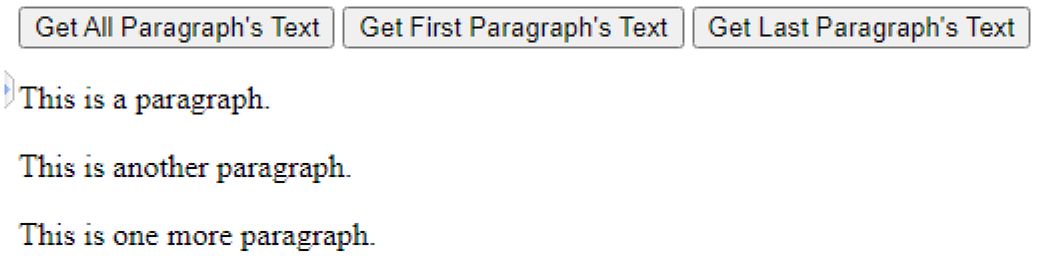
- The **attr()** method can be used to either fetch the value of an attribute from the first element in the matched set or set attribute values onto all matched elements.

Example

- Following is a simple example which fetches title attribute of <em> tag and set <div id = "divid"> value with the same value –

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="utf-8">
<title>jQuery Get Text Contents of the Elements</title>
<script src="https://code.jquery.com/jquery-3.5.1.min.js"></script>
<script>
$(document).ready(function(){
    $(".btn-one").click(function(){
```

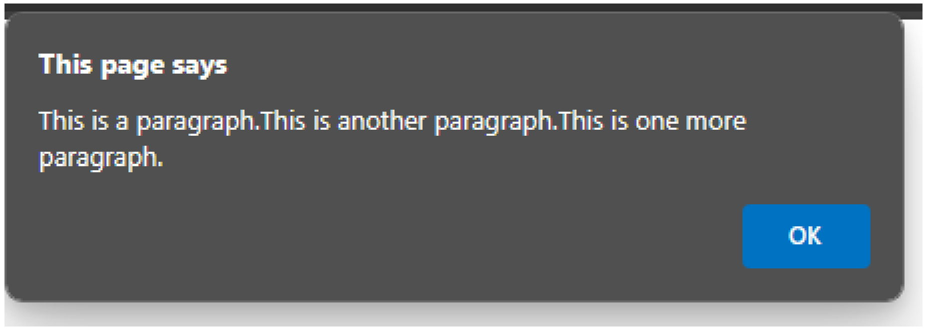
```
$(document).ready(function(){
    $(".btn-one").click(function(){
        var str = $("p").text();
        alert(str);
    });
    $(".btn-two").click(function(){
       var str = $("p:first").text();
       alert(str);
    });
    $(".btn-three").click(function(){
       var str = $("p.extra").text();
       alert(str);
    });
});
</script>
</head>
<body>
    <button type="button" class="btn-one">Get All Paragraph's Text</button>
    <button type="button" class="btn-two">Get First Paragraph's Text</button>
    <button type="button" class="btn-three">Get Last Paragraph's Text</button>
    <p>This is a paragraph.</p>
    <p>This is another paragraph.</p>
    <p class="extra">This is one more paragraph.</p>
</body>
</html>
```



This is a paragraph.

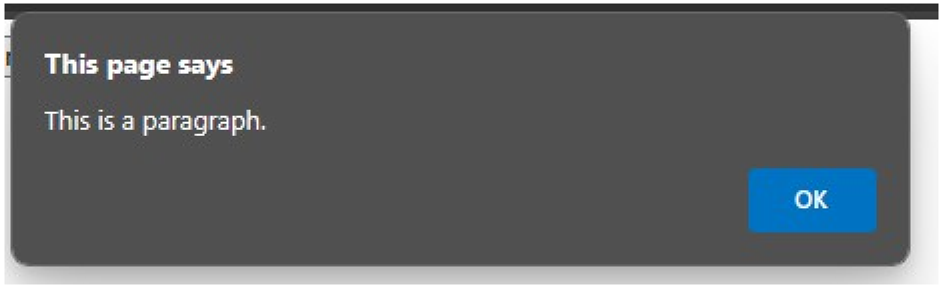
This is another paragraph.

This is one more paragraph.



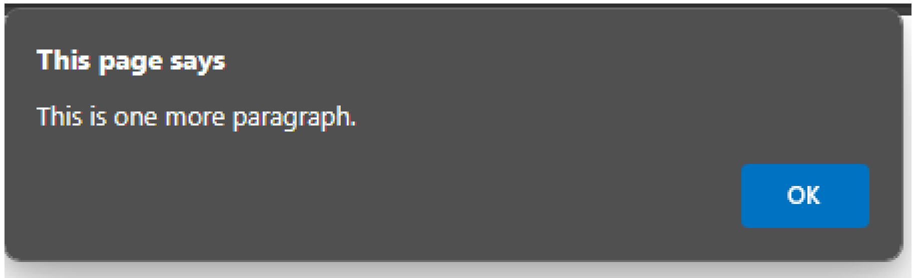
This page says

This is a paragraph.This is another paragraph.This is one more paragraph.



This page says

This is a paragraph.



This page says

This is one more paragraph.

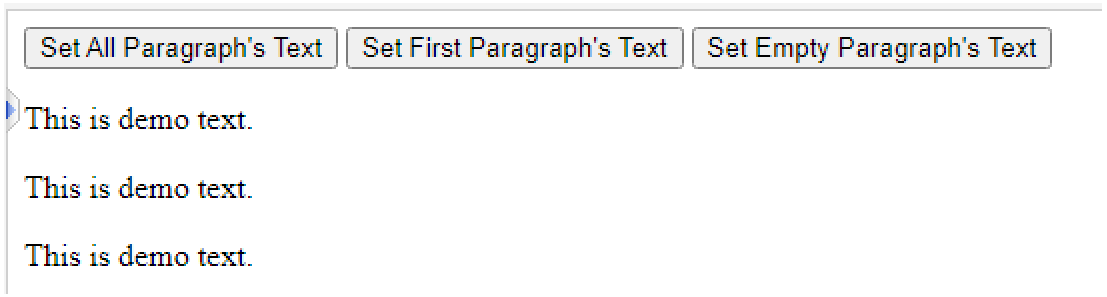
Set Attribute Value

- The **attr(name, value)** method can be used to set the named attribute onto all elements in the wrapped set using the passed value.

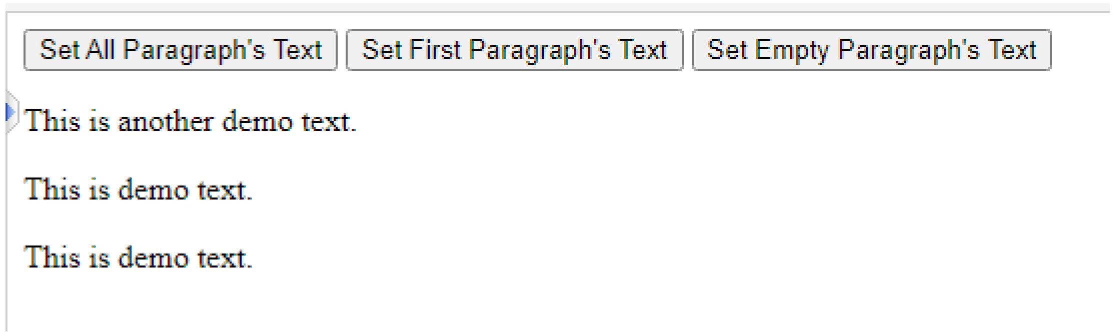
Example

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="utf-8">
<title>jQuery Set Text Contents of the Elements</title>
<script src="https://code.jquery.com/jquery-3.5.1.min.js"></script>
<script>
$(document).ready(function(){
    $(".btn-one").click(function(){
        $("p").text("This is demo text.");
    });
    $(".btn-two").click(function(){
        $("p:first").text("This is another demo text.");
    });
    $(".btn-three").click(function(){
        $("p.empty").text("This is one more demo text.");
    });
});
```

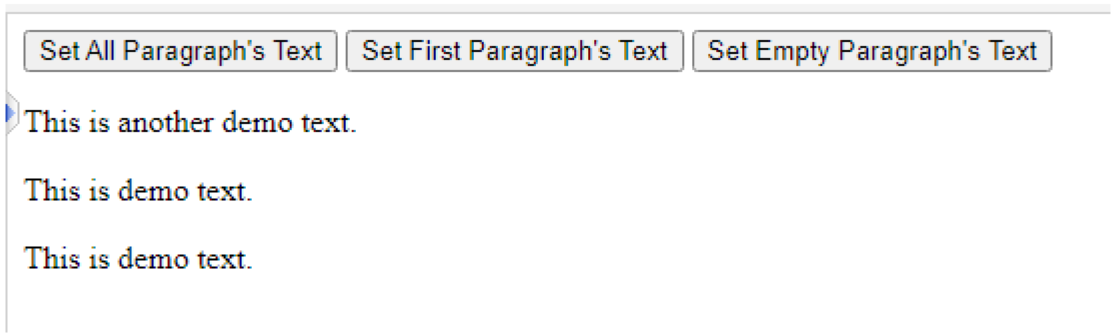
```
</script>
</head>
<body>
    <button type="button" class="btn-one">Set All Paragraph's Text</button>
    <button type="button" class="btn-two">Set First Paragraph's Text</button>
    <button type="button" class="btn-three">Set Empty Paragraph's Text</button>
    <p>This is a test paragraph.</p>
    <p>This is another test paragraph.</p>
    <p class="empty"></p>
</body>
</html>
```



```
This is demo text.

This is demo text.

This is demo text.
```



```
This is another demo text.

This is demo text.

This is demo text.
```



This is another demo text.

This is demo text.

This is demo text.

Applying Styles

- The **addClass(classes)** method can be used to apply defined style sheets onto all the matched elements. You can specify multiple classes separated by space.

Example

- Following is a simple example which sets **class** attribute of a para <p> tag –

```
<html>
  <head>
    <title>The jQuery Example</title>
    <script type = "text/javascript"
      src = "https://ajax.googleapis.com/ajax/libs/jquery/2.1.3/jquery.min.js">
    </script>
```

```
<script type = "text/javascript" language = "javascript">
      $(document).ready(function() {
        $("em").addClass("selected");
        $("#myid").addClass("highlight");
      });
    </script>

    <style>
      .selected { color:red; }
      .highlight { background:yellow; }
    </style>
  </head>

  <body>
    <em title = "Bold and Brave">This is first paragraph.</em>
    <p id = "myid">This is second paragraph.</p>
  </body>
</html>
```

Syntax:

- $(selector).css(property)
- $(selector).css(property, value)
- $(selector).css(property, function(index, currentvalue))\
- $(selector).css({property:value, property:value, ...})

jQuery Event Methods

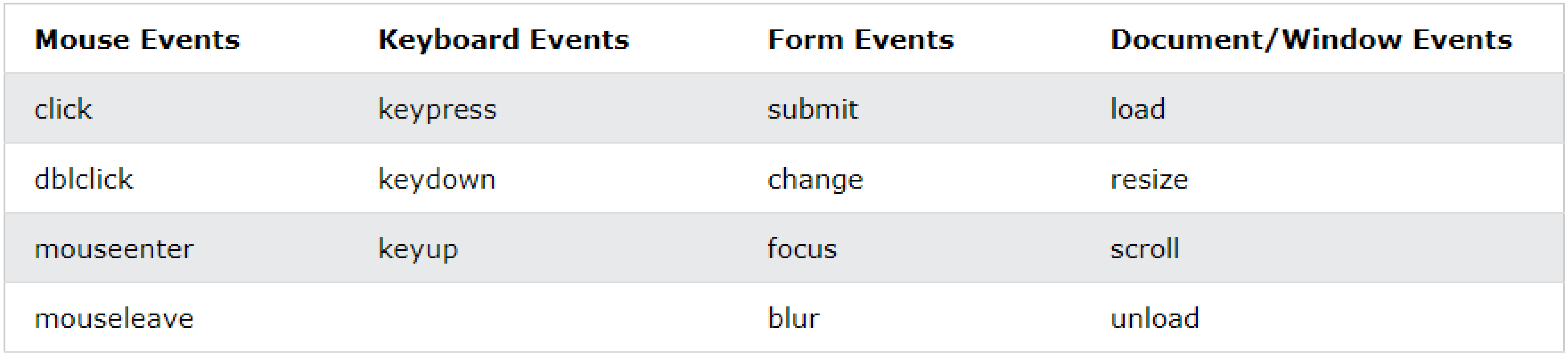
- All the different visitors' actions that a web page can respond to are called events.

Examples:

- moving a mouse over an element
- selecting a radio button
- clicking on an element

The term **"fires/fired"** is often used with events. Example: "The keypress event is fired, the moment you press a key".

Here are some common DOM events:



Mouse Events	Keyboard Events	Form Events	Document/Window Events
click	keypress	submit	load
dblclick	keydown	change	resize
mouseenter	keyup	focus	scroll
mouseleave		blur	unload

jQuery Syntax For Event Methods

- In jQuery, most DOM events have an equivalent jQuery method.
- To assign a click event to all paragraphs on a page, you can do this:

$("p").click();

Example:

```
$("p").click(function(){
  // action goes here!!
});
```

1. click()

- The click() method attaches an event handler function to an HTML element.
- The function is executed when the user clicks on the HTML element.
- The following example says: When a click event fires on a <p> element; hide the current <p> element:

```
<!DOCTYPE html>
<html>
<head>
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>
<script>
$(document).ready(function(){
  $("p").click(function(){
    $(this).hide();
  });
});
</script>
</head>
<body>
<p>If you click on me, I will disappear.</p>
<p>Click me away!</p>
<p>Click me too!</p>

</body>
</html>
```

mouseenter()

- The mouseenter() method attaches an event handler function to an HTML element.
- The function is executed when the mouse pointer enters the HTML element:

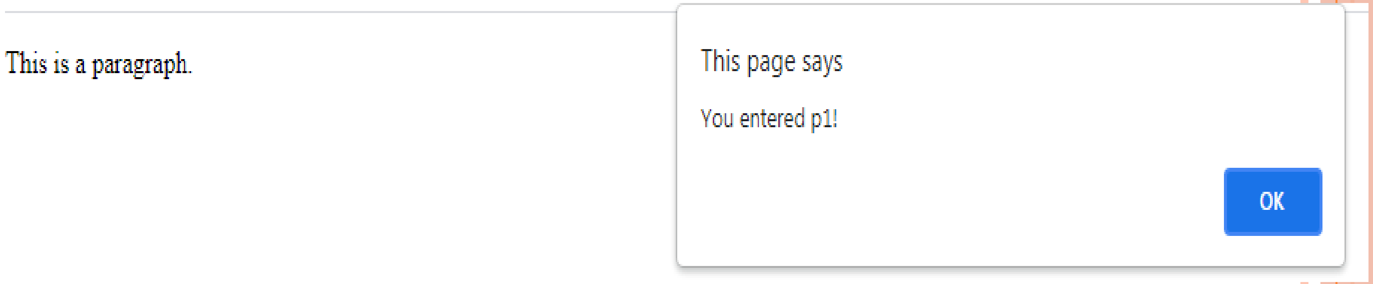
Example:

```
<!DOCTYPE html>
<html>
<head>
<script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>
<script>
$(document).ready(function(){
  $("#p1").mouseenter(function(){
    alert("You entered p1!");
  });
});
```

```
</script>
</head>
<body>
<p id="p1">Enter this paragraph.</p>
</body>
</html>
```

hover()

- The hover() method takes two functions and is a combination of the mouseenter() and mouseleave() methods.
- The first function is executed when the mouse enters the HTML element, and the second function is executed when the mouse leaves the HTML element:


This is a paragraph.

This page says

You entered p1!

Example:

```
<!DOCTYPE html>
<html>
<head>
<script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>
```

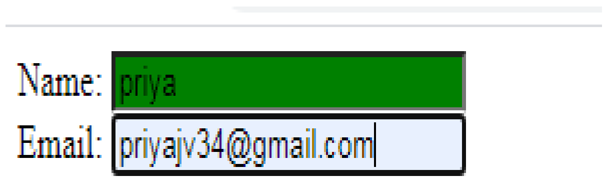
```
<script>
$(document).ready(function(){
  $("#p1").hover(function(){
    alert("You entered p1!");
  },
  function(){
    alert("Bye! You now leave p1!");
  });
});
</script>
</head>
<body>

<p id="p1">This is a paragraph.</p>

</body>
</html>
```

focus()

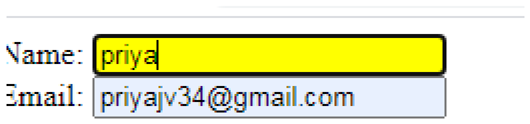
- The focus() method attaches an event handler function to an HTML form field.
- The function is executed when the form field gets focus:

Example:

```
<!DOCTYPE html>
<html>
<head>
<script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>
<script>
$(document).ready(function(){
 $("input").focus(function(){
  $(this).css("background-color", "yellow");
 });
 $("input").blur(function(){
  $(this).css("background-color", "green");
 });
});
```



Name: priya
Email: priyajv34@gmail.com



Name: priya
Email: priyajv34@gmail.com

JQUERY EFFECTS

jQuery hide() and show()

- With jQuery, you can hide and show HTML elements with the hide() and show() methods:

Example:

```
<!DOCTYPE html>
<html>
<head>
<script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>
<script>
$(document).ready(function(){
  $("#hide").click(function(){
    $("p").hide();
  });
  $("#show").click(function(){
    $("p").show();
  });
});
```

```
</script>
</head>
<body>
<p>If you click on the "Hide" button, It will disappear.</p>
<button id="hide">Hide</button>
<button id="show">Show</button>
</body>
</html>
```

Syntax:

- **$(*selector*).hide(*speed,callback*);**
- **$(*selector*).show(*speed,callback*);**

- The optional speed parameter specifies the speed of the hiding/showing, and can take the following values: "slow", "fast", or milliseconds.
- The optional callback parameter is a function to be executed after the hide() or show() method completes.

The following example demonstrates the speed parameter with hide():

```
<!DOCTYPE html>
<html>
<head>
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>
<script>
$(document).ready(function(){
  $("button").click(function(){
    $("p").hide(1000);
  });
});
</script>
</head>
<body>

<button>Hide</button>

<p>This is a paragraph with little content.</p>
<p>This is another small paragraph.</p>

</body>
</html>
```

jQuery Fading Methods

- With jQuery you can fade an element in and out of visibility.
- jQuery has the following fade methods:

jQuery fadeIn() Method

- The jQuery fadeIn() method is used to fade in a hidden element.

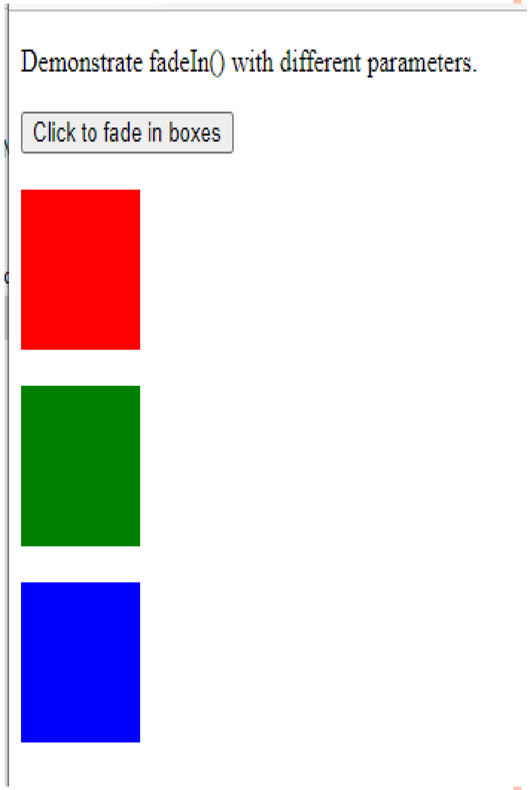
Syntax:

$(*selector*).fadeIn(*speed,callback*);

- The optional speed parameter specifies the duration of the effect. It can take the following values: "slow", "fast", or milliseconds.
- The optional callback parameter is a function to be executed after the fading completes.
- The following example demonstrates the fadeIn() method with different parameters:

```
<!DOCTYPE html>
<html>
<head>
<script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>
<script>
$(document).ready(function(){
  $("button").click(function(){
    $("#div1").fadeIn();
    $("#div2").fadeIn("slow");
    $("#div3").fadeIn(3000);
  });
});
</script>
</head>
<body>
```

```
<p>Demonstrate fadeIn() with different parameters.</p>

<button>Click to fade in boxes</button><br><br>

<div id="div1"

style="width:80px;height:80px;display:none;background-

color:red;"></div><br>
<div id="div2"

style="width:80px;height:80px;display:none;background-

color:green;"></div><br>
<div id="div3"

style="width:80px;height:80px;display:none;background-

color:blue;"></div>

</body>
</html>
```


Demonstrate fadeIn() with different parameters.

jQuery fadeOut() Method

- The jQuery fadeOut() method is used to fade out a visible element.

Syntax:

$(*selector*).fadeOut(*speed,callback*);

Example:

```
<!DOCTYPE html>
<html>
<head>
<script

src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.

1/jquery.min.js"></script>
<script>
$(document).ready(function(){
  $("button").click(function(){
    $("#div1").fadeOut();
    $("#div2").fadeOut("slow");
    $("#div3").fadeOut(3000);
  });
});
```

```
</script>
</head>
<body>
<p>Demonstrate fadeOut() with different parameters.</p>
<button>Click to fade out boxes</button><br><br>
<div id="div1"
style="width:80px;height:80px;background-color:red;"></div><br>
<div id="div2"
style="width:80px;height:80px;background-color:green;"></div><br>
<div id="div3"
style="width:80px;height:80px;background-color:blue;"></div>
</body>
</html>
```

jQuery fadeToggle() Method

- The jQuery fadeToggle() method toggles between the fadeIn() and fadeOut() methods.
- If the elements are faded out, fadeToggle() will fade them in.
- If the elements are faded in, fadeToggle() will fade them out.

Syntax:

$(*selector*).fadeToggle(*speed,callback*);

Example:

```
<html>
<head>
<script

src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>
<script>
$(document).ready(function(){
  $("button").click(function(){
```

```
 $("#div1").fadeToggle();
  $("#div2").fadeToggle("slow");
  $("#div3").fadeToggle(3000);
});
});
</script>
</head>
<body>
<p>Demonstrate fadeToggle() with different speed parameters.</p>
<button>Click to fade in/out boxes</button><br><br>
<div id="div1"
style="width:80px;height:80px;background-color:red;"></div>
<br>
<div id="div2"
style="width:80px;height:80px;background-color:green;"></div>
<br>
<div id="div3"
style="width:80px;height:80px;background-color:blue;"></div>
```

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

<u>jQuery fadeTo() Method</u>

- The jQuery fadeTo() method allows fading to a given opacity (value between 0 and 1).

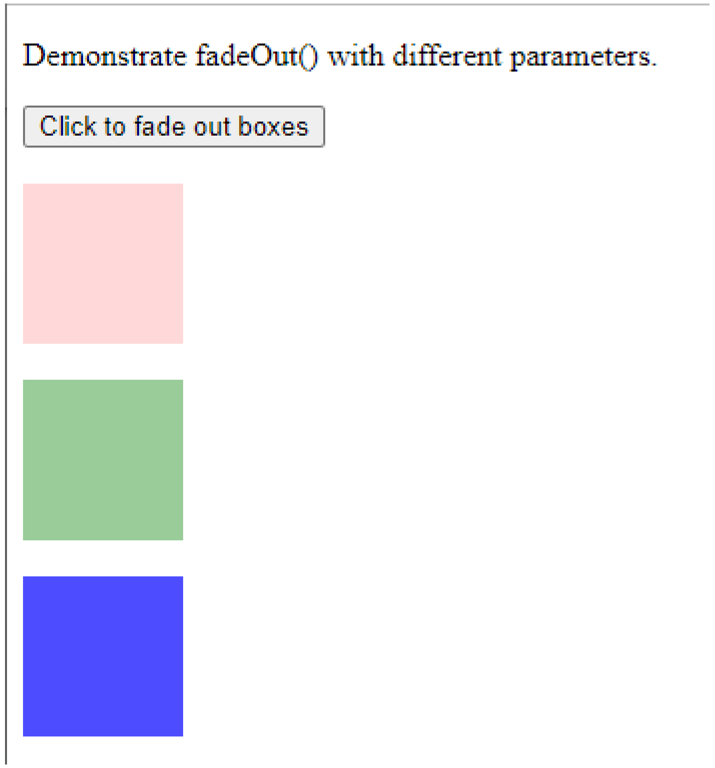
<u>Syntax:</u>

$(*selector*).fadeTo(*speed,opacity,callback*);

<u>Example:</u>

```
<!DOCTYPE html>
<html>
<head>
<script

src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>
<script>
```

```
$(document).ready(function(){
  $("button").click(function(){
    $("#div1").fadeTo("slow", 0.15);
    $("#div2").fadeTo("slow", 0.4);
    $("#div3").fadeTo("slow", 0.7);
  });
});
</script>
</head
<body>
<p>Demonstrate fadeTo() with different parameters.</p>
<button>Click to fade boxes</button><br><br>
<div id="div1"
style="width:80px;height:80px;background-color:red;"></div><br>>
<div id="div2"
style="width:80px;height:80px;background-color:green;"></div><br>
```


Demonstrate fadeOut() with different parameters.

```
<div id="div3"
style="width:80px;height:80px;backgroundcolor:blue;"></div>
</body>
</html>
```

jQuery Sliding Methods

With jQuery you can create a sliding effect on elements.

jQuery has the following slide methods:

- slideDown()
- slideUp()
- slideToggle()

<u>jQuery slideDown() Method</u>

The jQuery slideDown() method is used to slide down an element.

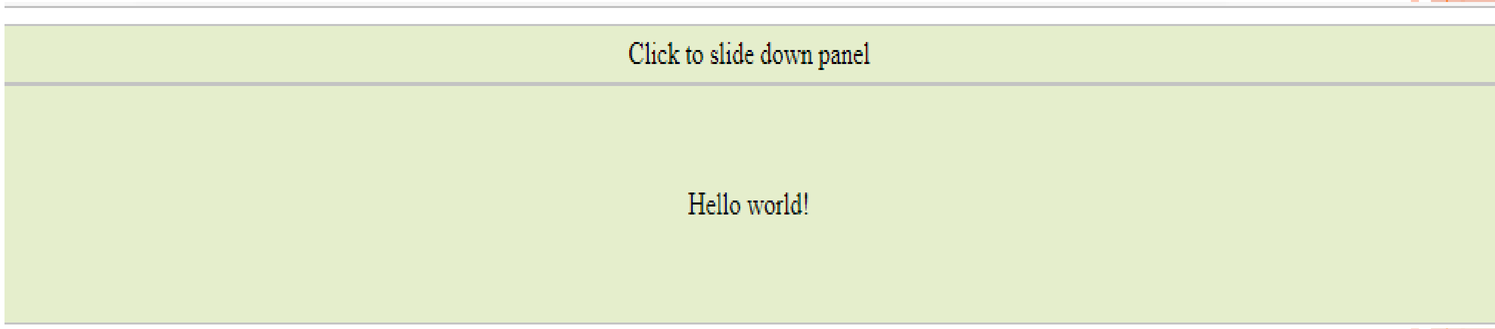
<u>Syntax:</u>

$(*selector*).slideDown(*speed,callback*);

<u>Example:</u>

```
<!DOCTYPE html>
<html>
<head>
<script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>
```

```
<script>
$(document).ready(function(){
  $("#flip").click(function(){
    $("#panel").slideDown("slow");
  });
});
</script>
<style>
#panel, #flip {
  padding: 5px;
  text-align: center;
  background-color: #e5eecc;
  border: solid 1px #c3c3c3;
}
#panel {
  padding: 50px;
  display: none;
}
```


Click to slide down panel

Hello world!

```
</style>
</head>
<body>

<div id="flip">Click to slide down panel</div>
<div id="panel">Hello world!</div>

</body>
</html>
```

jQuery slideUp() Method

The jQuery slideUp() method is used to slide up an element.

Syntax:

$(*selector*).slideUp(*speed,callback*);

Example:

```
<!DOCTYPE html>
<html>
<head>
<script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>
<script>
$(document).ready(function(){
  $("#flip").click(function(){
    $("#panel").slideUp("slow");
  });
});
<style>
#panel, #flip {
  padding: 5px;
  text-align: center;
  background-color: #e5eecc;
  border: solid 1px #c3c3c3;
}
```

```
#panel {
  padding: 50px;
}
</style>
</head>
<body>
 <div id="flip">Click to slide up panel</div>
<div id="panel">Hello world!</div>
</body>
</html>
```

jQuery slideToggle() Method

- The jQuery slideToggle() method toggles between the slideDown() and slideUp() methods.
- If the elements have been slid down, slideToggle() will slide them up.
- If the elements have been slid up, slideToggle() will slide them down.

Example:

```
<!DOCTYPE html>
<html>
<head>
<script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>
<script>
$(document).ready(function(){
  $("#flip").click(function(){
    $("#panel").slideToggle("slow");
  });
});
</script>
<style>
#panel, #flip {
  padding: 5px;
  text-align: center;
  background-color: #e5eecc;
  border: solid 1px #c3c3c3;
}
```

```
#panel {
  padding: 50px;
  display: none;
}
</style>
</head>
<body>

<div id="flip">Click to slide the panel down or

up</div>
<div id="panel">Hello world!</div>

</body>
</html>
```

jQuery Effects - Animation

The animate() Method

The jQuery animate() method is used to create custom animations.

Syntax:

$(*selector*).animate({*params*},*speed,callback*);

- The required params parameter defines the CSS properties to be animated.

The following example demonstrates a simple use of the animate() method;
it moves a <div> element to the right, until it has reached a left property of 250px:

EXAMPLE:

```
<!DOCTYPE html>
<html>
<head>
<script

src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>
```

```
<script>
$(document).ready(function(){
$("button").click(function(){
$("div").animate({
left: '250px',
opacity: '0.5',
height: '150px',
width: '150px'
});
});
});
</script>
</head>
<body>
<button>start animation</button>
<div style="background:#98bf21;height:100px;width:100px;position:absolute;">
</div>
</body>
</html>
```

NOTE:

By default, all HTML elements have a static position, and cannot be moved. To manipulate the position, remember to first set the CSS position property of the element to relative, fixed, or absolute!

jQuery stop() Method

- The jQuery stop() method is used to stop an animation or effect before it is finished.
- The stop() method works for all jQuery effect functions, including sliding, fading and custom animations.

Syntax:

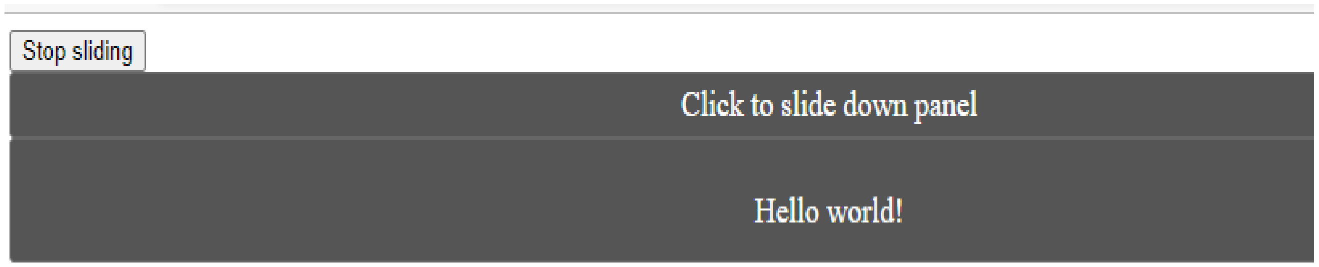
$(*selector*).stop(*stopAll,goToEnd*);

EXAMPLE:

```
<!DOCTYPE html>
<html>
<head>
<script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>
<script>
```

```
$(document).ready(function(){
  $("#flip").click(function(){
    $("#panel").slideDown(5000);
  });
  $("#stop").click(function(){
    $("#panel").stop();
  });
});
</script>
<style>
#panel, #flip {
  padding: 5px;
  font-size: 18px;
  text-align: center;
  background-color: #555;
  color: white;
  border: solid 1px #666;
  border-radius: 3px;
}
```

```
#panel {
  padding: 50px;
  display: none;
}
</style>
</head>
<body>

<button id="stop">Stop sliding</button>

<div id="flip">Click to slide down panel</div>
<div id="panel">Hello world!</div>

</body>
</html>
```



Stop sliding

Click to slide down panel

Hello world!

jQuery Callback Functions

- JavaScript statements are executed line by line. However, with effects, the next line of code can be run even though the effect is not finished. This can create errors.
- To prevent this, you can create a callback function.
- A callback function is executed after the current effect is finished.
- Typical syntax: **$(*selector*).hide(*speed,callback*);**

  The example below has a callback parameter that is a function that will be executed after the hide effect is completed:

Example with Callback

```
<!DOCTYPE html>
<html>
<head>
<script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>
<script>
```

```
$(document).ready(function(){
  $("button").click(function(){
    $("p").hide("slow", function(){
      alert("The paragraph is now hidden");
    });
  });
});
</script>
</head>
<body>

<button>Hide</button>

<p>This is a paragraph with little content.</p>

</body>
</html>
```

Example without Callback

```
<!DOCTYPE html>
<html>
<head>
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>
<script>
$(document).ready(function(){
  $("button").click(function(){
    $("p").hide(1000);
    alert("The paragraph is now hidden");
  });
});
</script>
</head>
<body>
<button>Hide</button>
<p>This is a paragraph with little content.</p>
</body>
</html>
```

jQuery - Chaining

- With jQuery, you can chain together actions/methods.
- Chaining allows us to run multiple jQuery methods (on the same element) within a single statement.
- It allows us to run multiple jQuery commands, one after the other, on the same element(s).

  The following example chains together the css(), slideUp(), and slideDown() methods. The "p1" element first changes to red, then it slides up, and then it slides down:

Example:

```
<!DOCTYPE html>
<html>
<head>
<script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>
    <script>
$(document).ready(function(){
  $("button").click(function(){
$("#p1").css("color", "red").slideUp

(2000).slideDown(2000);
  });
});
</script>
</head>
```

```
<body>

<p id="p1">jQuery is fun!!</p>

<button>Click me</button>

</body>
</html>
```

Traversing the DOM

- jQuery provides a variety of methods that allow us to traverse the DOM.
- The largest category of traversal methods are tree-traversal.
- jQuery traversing, which means "move through", are used to "find" (or select) HTML elements based on their relation to other elements.
- With jQuery you can traverse up the DOM tree to find ancestors of an element.
- An ancestor is a parent, grandparent, great-grandparent, and so on.

Traversing Up the DOM Tree

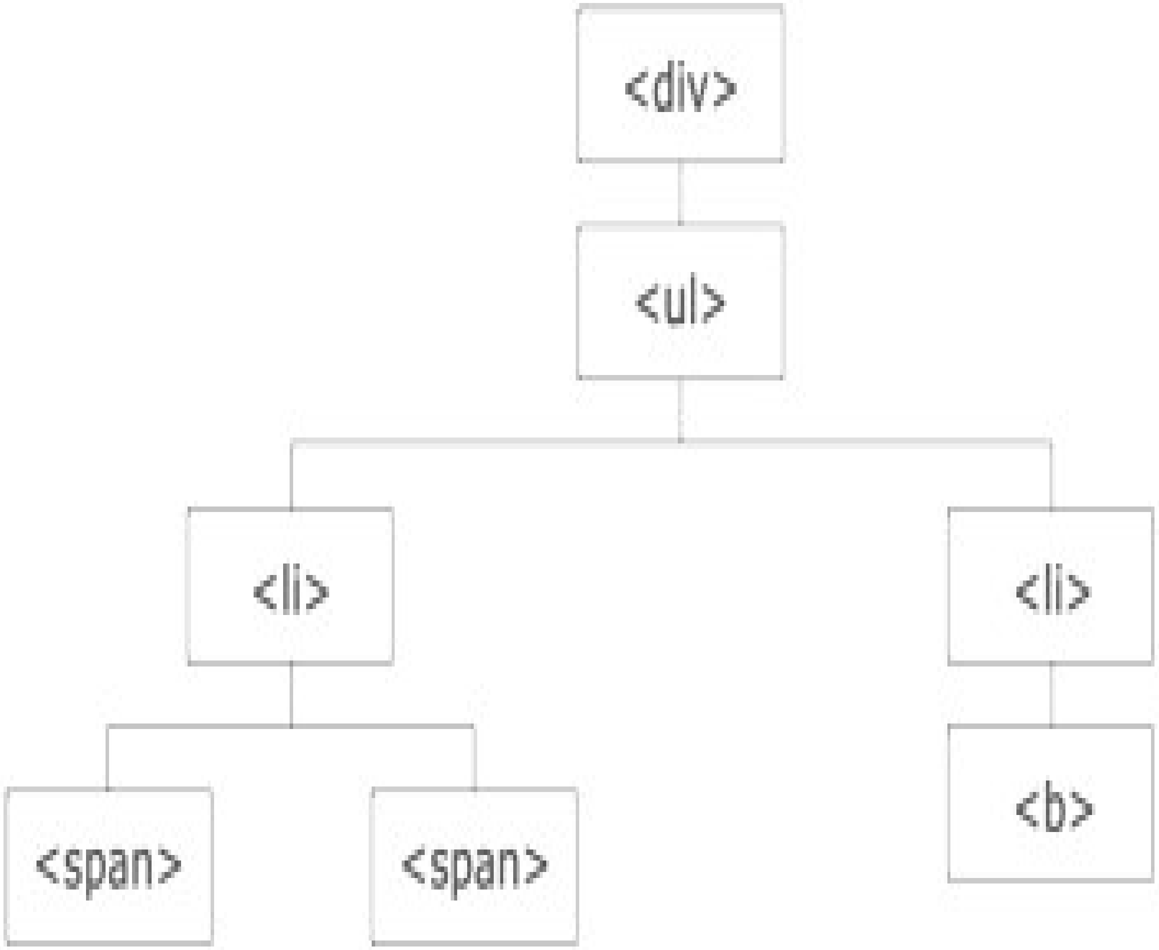
Three useful jQuery methods for traversing up the DOM tree are:

- parent()
- parents()
- parentsUntil()

jQuery parent() Method

- The parent() method returns the direct parent element of the selected element.
- This method only traverse a single level up the DOM tree. The following example returns the direct parent element of each <span> elements:

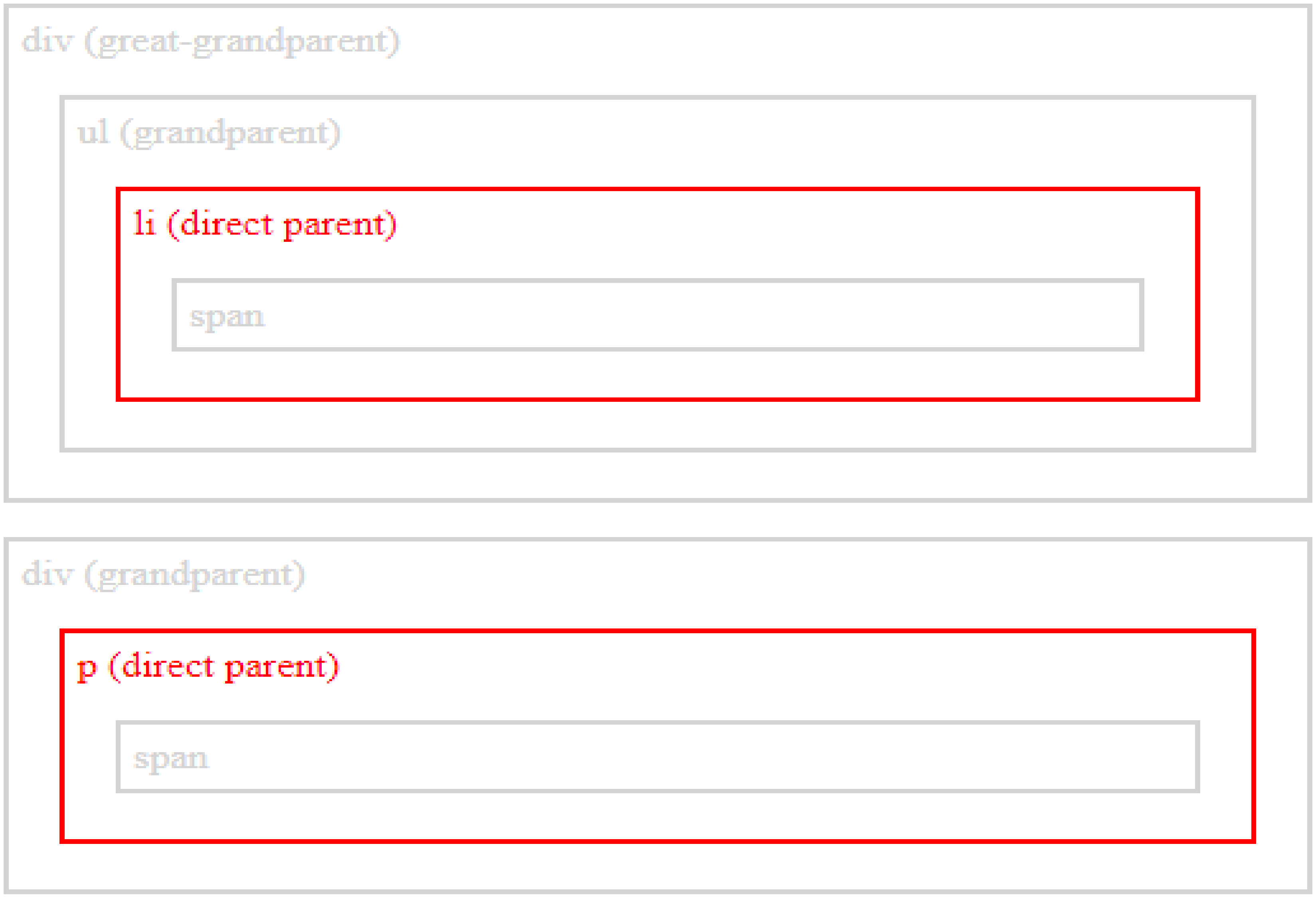
```
html
    -- head
        -- meta
        -- title
        -- script
    -- body
        -- div
            -- h1
            -- p
                -- em
            -- ul
                -- li
                -- li
```



Example:

```
<!DOCTYPE html>
<html>
<head>
<style>
.ancestors * {
  display: block;
  border: 2px solid lightgrey;
  color: lightgrey;
  padding: 5px;
  margin: 15px;
}
</style>
<script>
src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>
<script
$(document).ready(function(){
  $("span").parent().css({"color": "red", "border": "2px solid red"});
});
```

```
</script>
</head>
<body>

<div class="ancestors">
  <div style="width:500px;">div (great-grandparent)
    <ul>ul (grandparent)
      <li>li (direct parent)
        <span>span</span>
      </li>
    </ul>
  </div>

  <div style="width:500px;">div (grandparent)
    <p>p (direct parent)
      <span>span</span>
    </p>
  </div>
</div>

</body>
</html>
```

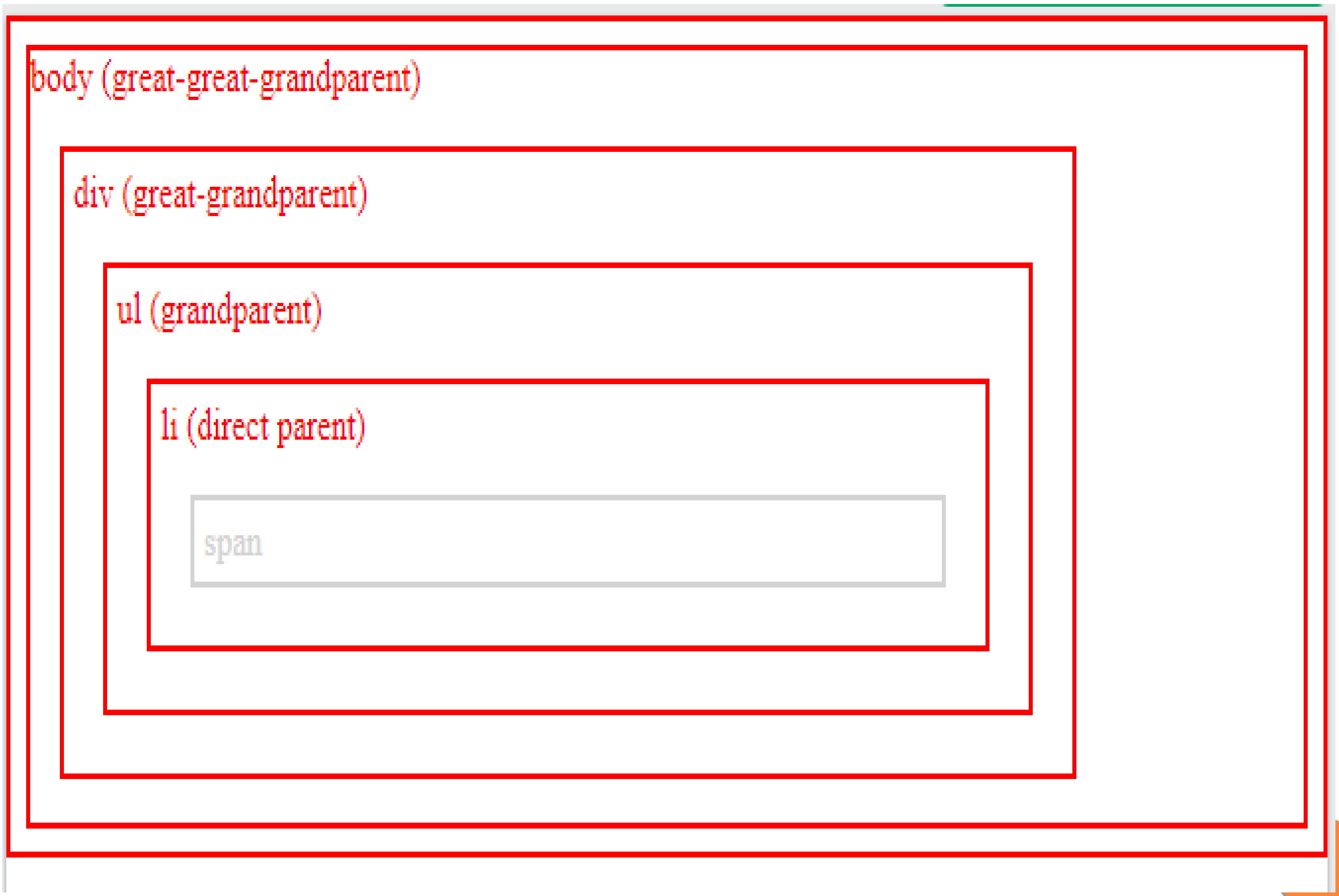

div (great-grandparent)
ul (grandparent)
li (direct parent)
span
div (grandparent)
p (direct parent)
span

jQuery parents() Method

- The parents() method returns all ancestor elements of the selected element, all the way up to the document's root element (<html>).
- The following example returns all ancestors of all <span> elements:

```
<!DOCTYPE html>
<html>
<head>
<style>
.ancestors * {
  display: block;
  border: 2px solid lightgrey;
  color: lightgrey;
  padding: 5px;
  margin: 15px;
}
</style>
<script

src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>
```

```
<script>
$(document).ready(function(){
  $("span").parents().css({"color": "red", "border": "2px solid red"});
});
</script>
</head>

<body class="ancestors">body (great-great-grandparent)
  <div style="width:500px;">div (great-grandparent)
    <ul>ul (grandparent)
      <li>li (direct parent)
        <span>span</span>
      </li>
    </ul>
  </div>
</body>

<!-- The outer red border, before the body element, is the html element (also an
   ancestor) -->
</html>
```

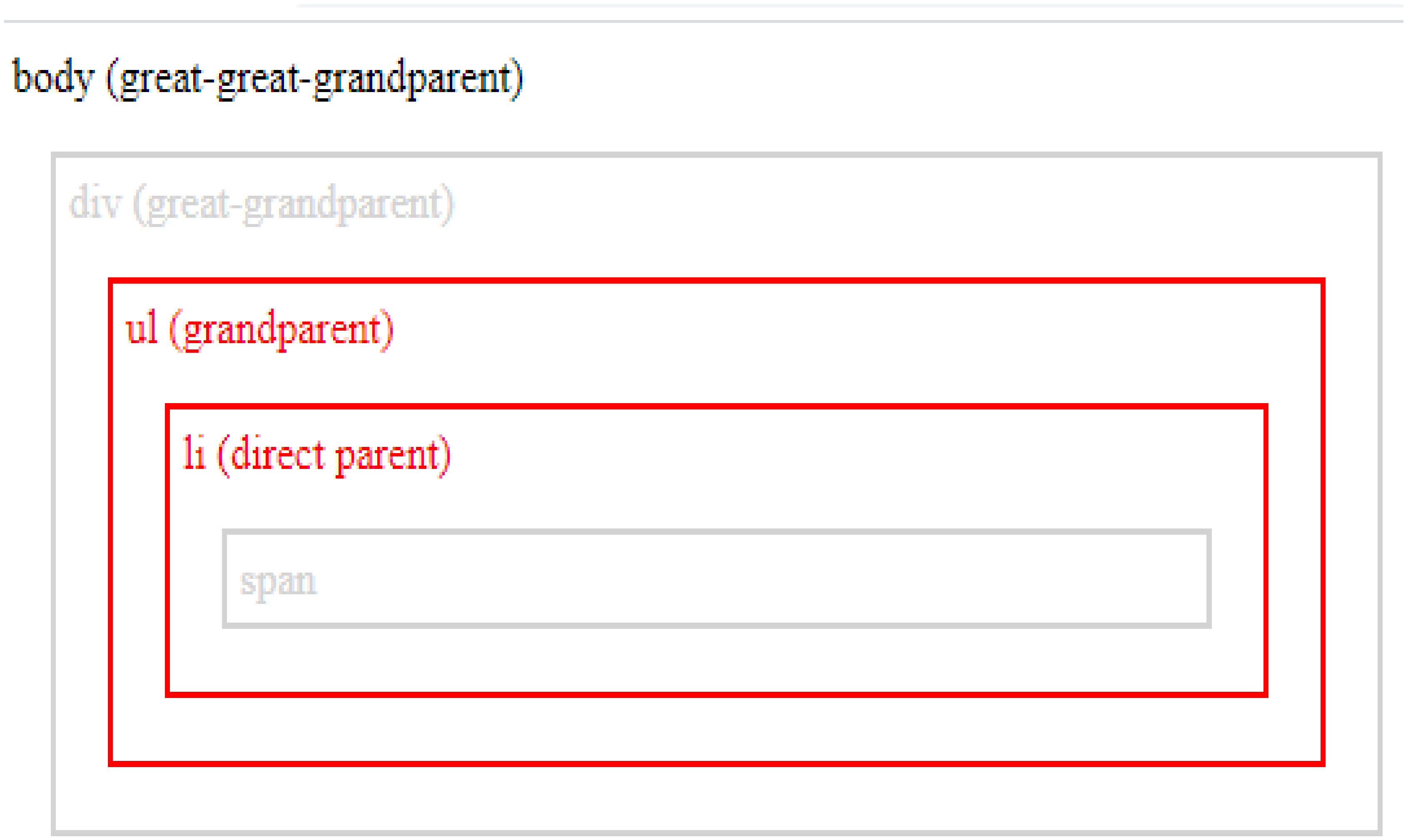

body (great-great-grandparent)
div (great-grandparent)
ul (grandparent)
li (direct parent)
span

jQuery parentsUntil() Method

- The parentsUntil() method returns all ancestor elements between two given arguments.
- The following example returns all ancestor elements between a <span> and a <div> element:

```
<!DOCTYPE html>
<html>
<head>
<style>
.ancestors * {
  display: block;
  border: 2px solid lightgrey;
  color: lightgrey;
  padding: 5px;
  margin: 15px;
}
</style>
```

```
<script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>
<script>
$(document).ready(function(){
  $("span").parentsUntil("div").css({"color": "red", "border": "2px solid red"});
});
</script>
</head>
<body class="ancestors"> body (great-great-grandparent)
  <div style="width:500px;">div (great-grandparent)
    <ul>ul (grandparent)
      <li>li (direct parent)
        <span>span</span>
      </li>
    </ul>
  </div>
</body>

</html>
```


body (great-great-grandparent)
div (great-grandparent)
ul (grandparent)
li (direct parent)
span

jQuery Traversing - Descendants

- With jQuery you can traverse down the DOM tree to find descendants of an element.
- A descendant is a child, grandchild, great-grandchild, and so on.

Traversing Down the DOM Tree

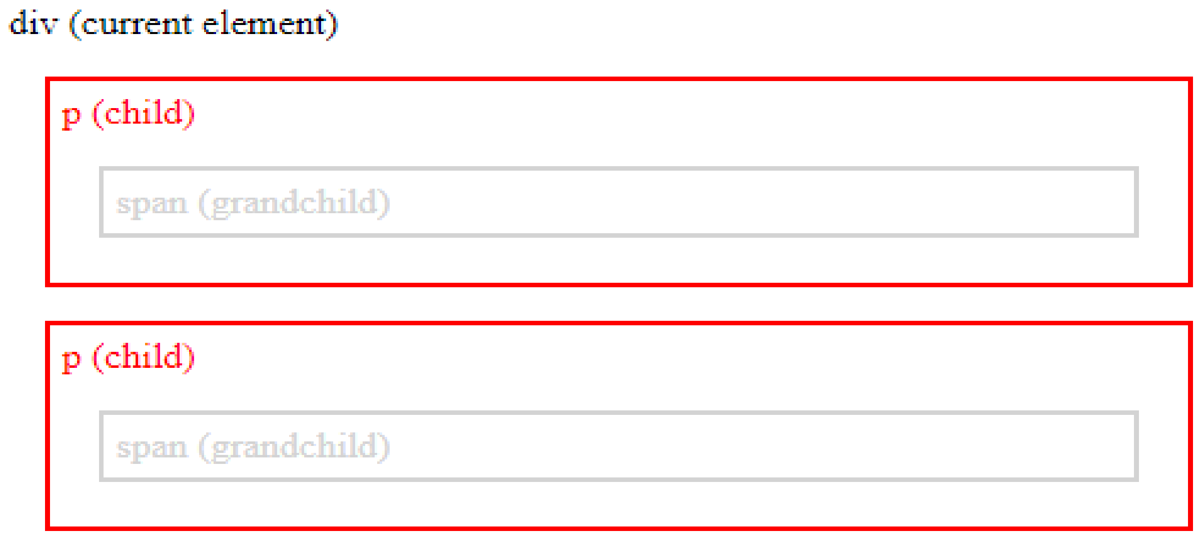
- Two useful jQuery methods for traversing down the DOM tree are:
- children()
- find()

jQuery children() Method

- The children() method returns all direct children of the selected element.
- This method only traverses a single level down the DOM tree.
- The following example returns all elements that are direct children of each <div> elements:

```
<!DOCTYPE html>
<html>
<head>
<style>
.descendants * {
  display: block;
  border: 2px solid lightgrey;
  color: lightgrey;
  padding: 5px;
  margin: 15px;
}
</style>
<script>
src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>
<script>
$(document).ready(function(){
  $("div").children().css({"color": "red", "border": "2px solid red"});
});
```

```
</script>
</head>
<body>
<div class="descendants" style="width:500px;">div (current element)
  <p>p (child)
    <span>span (grandchild)</span>
  </p>
  <p>p (child)
    <span>span (grandchild)</span>
  </p>
</div>
</body>
</html>
```

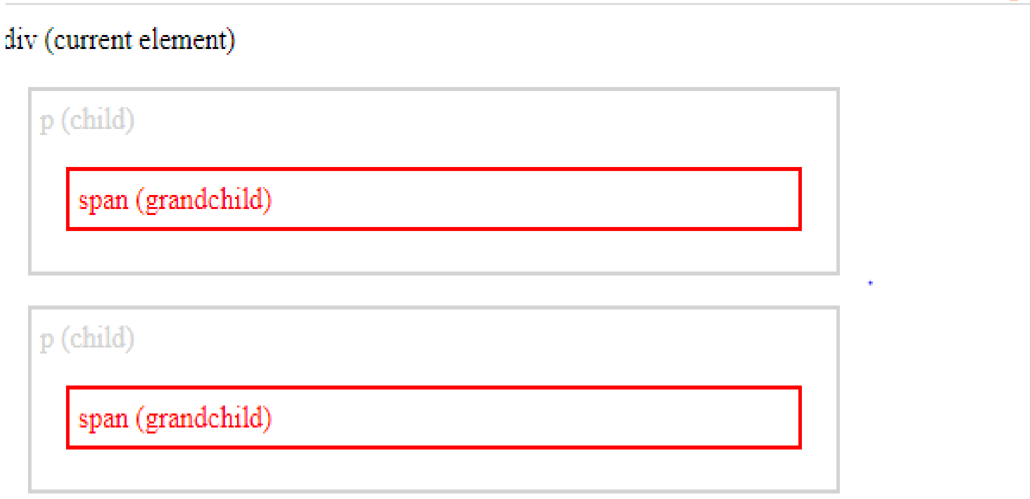


jQuery find() Method

- The find() method returns descendant elements of the selected element, all the way down to the last descendant.
- The following example returns all <span> elements that are descendants of <div>:

```
<!DOCTYPE html>
<html>
<head>
<style>
.descendants * {
  display: block;
  border: 2px solid lightgrey;
  color: lightgrey;
  padding: 5px;
  margin: 15px;
}
</style>
```

```
<script

src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>
<script>
$(document).ready(function(){
  $("div").find("span").css({"color": "red", "border": "2px solid red"});
});
</script>
</head>
<body>
<div class="descendants" style="width:500px;">div (current element)
  <p>p (child)
    <span>span (grandchild)</span>
  </p>
  <p>p (child)
    <span>span (grandchild)</span>
  </p>
</div>
</body>
</html>
```



```
div (current element)
p (child)
span (grandchild)
p (child)
span (grandchild)
```

- With jQuery you can traverse sideways in the DOM tree to find siblings of an element.
- Siblings share the same parent.

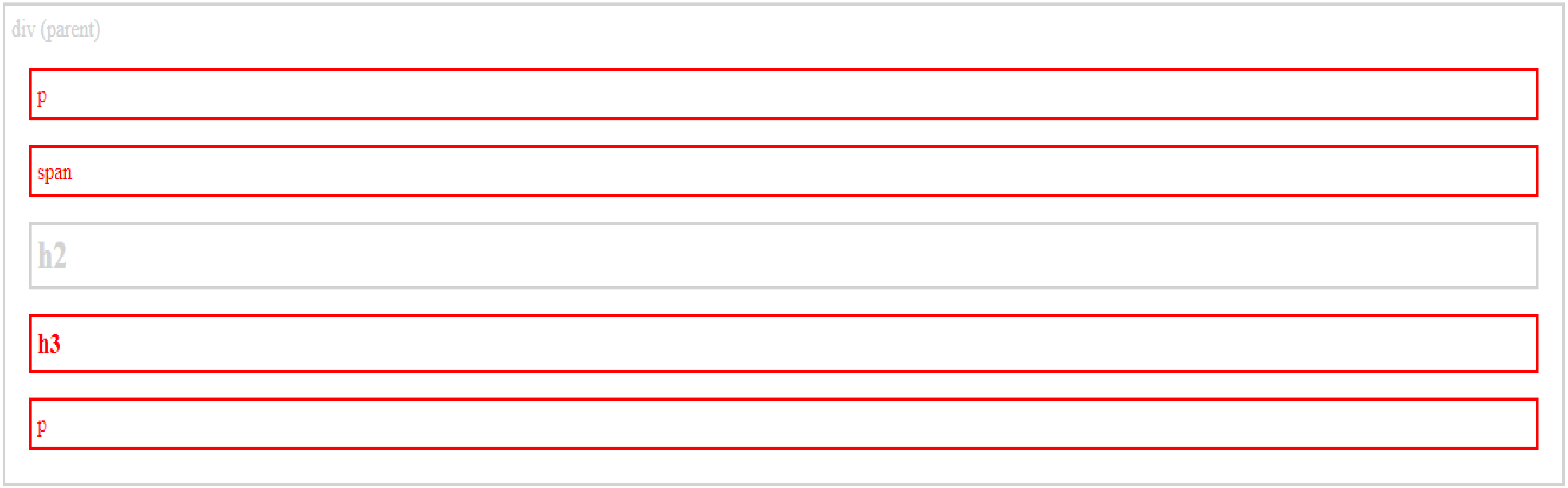
<u>Traversing Sideways in The DOM Tree</u>

<u>jQuery siblings() Method</u>

- The siblings() method returns all sibling elements of the selected element.
- The following example returns all sibling elements of <h2>:

```
<!DOCTYPE html>
<html>
<head>
<style>
.siblings * {
  display: block;
  border: 2px solid lightgrey;
  color: lightgrey;
  padding: 5px;
  margin: 15px;
}
</style>
```

```
<script

src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>
<script>
$(document).ready(function(){
  $("h2").siblings().css({"color": "red", "border": "2px solid red"});
});
</script>
</head>
<body class="siblings">

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

</body>
</html>
```


div (parent)
p
span
h2
h3
p

jQuery Traversing - Filtering

- The first(), last(), eq(), filter() and not() Methods
- The most basic filtering methods are first(), last() and eq(), which allow you to select a specific element based on its position in a group of elements.
- Other filtering methods, like filter() and not() allow you to select elements that match, or do not match, a certain criteria.

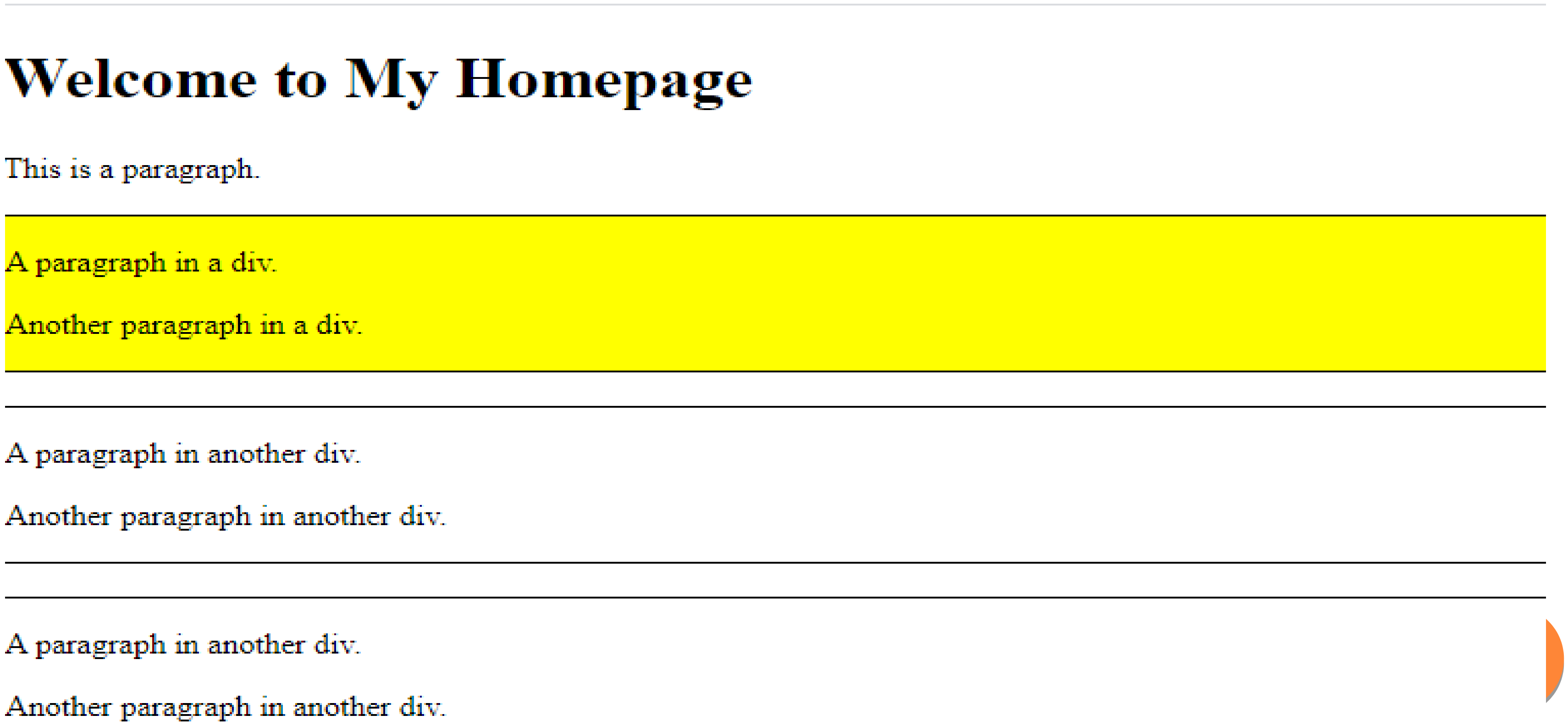
jQuery first() Method

- The first() method returns the first element of the specified elements.
- The following example selects the first <div> element:

```
<!DOCTYPE html>
<html>
<head>
<script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>
<script>
```

```
$(document).ready(function(){
  $("div").first().css("background-color", "yellow");
});
</script>
</head>
<body>

<h1>Welcome to My Homepage</h1>

<p>This is a paragraph.</p>

<div style="border: 1px solid black;">
  <p>A paragraph in a div.</p>
  <p>Another paragraph in a div.</p>
</div>
<br>

<div style="border: 1px solid black;">
  <p>A paragraph in another div.</p>
  <p>Another paragraph in another div.</p>
</div>
<br>
```

```
<div style="border: 1px solid black;">
  <p>A paragraph in another div.</p>
  <p>Another paragraph in another div.</p>
</div>
</body>
</html>
```



Welcome to My Homepage

This is a paragraph.

A paragraph in a div.

Another paragraph in a div.

A paragraph in another div.

Another paragraph in another div.

A paragraph in another div.

Another paragraph in another div.

JQUERY PLUGIN:

- A plug-in is piece of code written in a standard JavaScript file. These files provide useful jQuery methods which can be used along with jQuery library methods.
- There are plenty of jQuery plug-in available which you can download from repository link at https://jquery.com/plugins.

HOW TO USE PLUGINS:

- To make a plug-in's methods available to us, we include plug-in file very similar to jQuery library file in the <head> of the document.
- We must ensure that it appears after the main jQuery source file, and before our custom JavaScript code.

HOW TO DEVELOP A PLUG-IN:

- the syntax to create a method −

  jQuery.fn.methodName = methodDefinition;

- Any methods or functions you attach must have a semicolon (;) at the end.
- Your method must return the jQuery object, unless explicity noted otherwise.
- You should use this.each to iterate over the current set of matched elements - it produces clean and compatible code that way.
- Prefix the filename with jQuery, follow that with the name of the plugin, and conclude with .js.
- Always attach the plugin to jQuery directly instead of $, so users can use a custom alias via the noConflict() method.

WHAT IS GIT?

- It is a free, high-quality distributed version control system suitable for tracking modifications in source code in software development. It was originally created as an open-source system for coordinating tasks among programmers, but today it is widely used to track changes in any set of files. The key objectives of Git are as follows:
  - Speed and efficiency
  - Data integrity
  - Support for distributed and non-linear workflows

WHAT IS GITHUB?

- It is a web-based Git repository. This hosting service has cloud-based storage.
- GitHub offers all distributed version control and source code management functionality of Git while adding its own features.
- It makes it easier to collaborate using Git.
- Additionally, GitHub repositories are open to the public.
- Developers worldwide can interact and contribute to one another’s code, modify or improve it, making GitHub a networking site for web professionals.
- The process of interaction and contribution is also called social coding.

git	GitHub
1. It is a software	1. It is a service
2. It is installed locally on the system	2. It is hosted on Web
3. It is a command line tool	3. It provides a graphical interface
4. It is a tool to manage different versions of edits, made to files in a git repository	4. It is a space to upload a copy of the **Git** repository
5. It provides functionalities like Version Control System Source Code Management	5. It provides functionalities of Git like VCS, Source Code Management as well as adding few of its own features