jQuery

Contents at a Glance

Table of Contents

1. Introduction to jQuery

1.1. What is jQuery?

jQuery is a lightweight, "write less, do more", JavaScript library. The purpose of jQuery is to make it much easier to use JavaScript on your website. jQuery takes a lot of common tasks that require many lines of JavaScript code to accomplish, and wraps them into methods that you can call with a single line of code. jQuery also simplifies a lot of the complicated things from JavaScript, like AJAX calls and DOM manipulation. The jQuery library contains the following features:

- HTML/DOM manipulation
- CSS manipulation
- HTML event methods
- Effects and animations
- AJAX
- Utilities

Note : In addition, jQuery has plugins for almost any task out there.JavaScript Variables

1.2. Adding jQuery to Your Web Page

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

1.2.1. 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):

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <meta http-equiv="X-UA-Compatible" content="IE=edge" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <title>Hello World from jQuery</title>
  </head>
  <body>
    <p id="note">Hello, World!</p>

    <script src="jquery.js"></script>
    <script>
      $("#note").html("Hello World from jQuery!");
      $("#note").css({
        color: "red",
        "background-color": "pink",
        padding: "10px",
      });
    </script>
  </body>
</html>
```

1.2.2. jQuery CDN

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

1.3. 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".

1.3.1. Document Ready Event

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.

1.4. jQuery Selector

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. It's based on the existing CSS Selectors, and in addition, it has some own custom selectors. All selectors in jQuery start with the dollar sign and parentheses: $().

1.4.1. 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> </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>
    <script src="jquery.js"></script>
    <script>
      $(document).ready(function () {
        $("button").click(function () {
          $("p").hide();
        });
      });
    </script>
  </body>
</html>
```

1.4.2. 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:

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

1.4.3. 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:

```
<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>
<script src="jquery.js"></script>
<script>
$(document).ready(function(){
  $("button").click(function(){
    $(".test").hide();
  });
});
</script>
</body>
```

2. jQuery Effects

2.1.1. Hide and Show

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

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 (you will learn more about callback functions in a later chapter).The following example demonstrates the speed parameter with hide():

```
<body>
    <p>If you click on the "Hide" button, I will disappear.</p>
    <button id="hide">Hide</button>
    <button id="show">Show</button>
    <script src="jquery.js"></script>
    <script>
      $(document).ready(function () {
        $("#hide").click(function () {
          $("p").hide();
        });
        $("#show").click(function () {
          $("p").show();
        });
      });
    </script>
  </body>
```

2.1.2. jQuery Fading Methods

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

- fadeIn()
- fadeOut()
- fadeToggle()

- fadeTo()

(1) 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:

```
<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>
  <script src="jquery.js"></script>
  <script>
    $(document).ready(function () {
      $("button").click(function () {
        $("#div1").fadeIn();
        $("#div2").fadeIn("slow");
        $("#div3").fadeIn(3000);
      });
    });
  </script>
</body>
```

(2) jQuery FadeOut() Method

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

Syntax:

$(selector).fadeOut(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 fadeOut() method with different parameters:

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

(3) 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);

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 fadeToggle() method with different parameters:

```
<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>
  <script src="jquery.js"></script>
  <script>
    $(document).ready(function () {
      $("button").click(function () {
        $("#div1").fadeToggle();
        $("#div2").fadeToggle("slow");
        $("#div3").fadeToggle(3000);
      });
    });
  </script>
</body>
```

2.1.3. jQuery Sliding Methods

With jQuery you can create a sliding effect on elements. jQuery has the following slide methods:

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

(1) jQuery slideDown() Method

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

Syntax:

$(selector).slideDown(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 sliding completes. The following example demonstrates the slideDown() method:

```
<head>
  <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 down panel</div>
  <div id="panel">Hello world!</div>
  <script src="jquery.js"></script>
  <script>
    $(document).ready(function () {
      $("#flip").click(function () {
        $("#panel").slideDown("slow");
      });
    });
  </script>
</body></html>
```

(2) jQuery slideUp() Method

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

Syntax:

$(selector).slideUp(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 sliding completes. The following example demonstrates the slideUp() method:

```
<head>
  <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>
  <script src="jquery.js"></script>
  <script>
    $(document).ready(function () {
      $("#flip").click(function () {
        $("#panel").slideUp("slow");
      });
    });
  </script>
</body>
```

(3) 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.

$(selector).slideToggle(speed, callback);

The optional speed parameter can take the following values: "slow", "fast", milliseconds. The optional callback parameter is a function to be executed after the sliding completes. The following example demonstrates the slideToggle() method:

```
<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>
<script src="jquery.js"></script>
<script>
$(document).ready(function(){
  $("#flip").click(function(){
    $("#panel").slideToggle("slow");
  });
});
</script>
</body>
<div id="flip">Click to slide the panel down or up</div>
<div id="panel">Hello world!</div>
</body></html>
```

2.1.4. The animation() 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 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 animation completes.

```
<body>
  <p>Click the button multiple times to toggle the animation.</p>
  <button>Start Animation</button>
  <div
style="background:#98bf21;height:100px;width:100px;position:absolute;">
</div>
  <script src="jquery.js"></script>
  <script>
    $(document).ready(function () {
      $("button").click(function () {
        $("div").animate({
          height: 'toggle'
        });
      });
    });
  </script>
</body>
```

```
<body>
  <button>Start Animation</button>
  <div
style="background:#98bf21;height:100px;width:200px;position:absolute;">
HELLO</div>
  <script src="jquery.js"></script>
  <script>
    $(document).ready(function () {
      $("button").click(function () {
        var div = $("div");
        div.animate({ left: '100px' }, "slow");
        div.animate({ fontSize: '3em' }, "slow");
      });
    });
  </script>
</body>
```

2.1.5. jQuery Callback Function

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.

Syntax:

$(selector).hide(speed,callback);

```
<body>
<button>Hide</button>
<p>This is a paragraph with little content.</p>
<script src="jquery.js"></script>
<script>
$(document).ready(function(){
  $("button").click(function(){
    $("p").hide(1000);
    alert("The paragraph is now hidden");
  });
});
</script></body>
```

2.1.6. jQuery Method Chaining

Until now we have been writing jQuery statements one at a time (one after the other). However, there is a technique called chaining, that allows us to run multiple jQuery commands, one after the other, on the same element(s).

Note: This way, browsers do not have to find the same element(s) more than once.

```
<body>
  <p id="p1">jQuery is fun!!</p>
  <button>Click me</button>
  <script src="jquery.js"></script>
  <script>
    $(document).ready(function () {
      $("button").click(function () {
        $("#p1").css("color", "red")
          .slideUp(2000)
          .slideDown(2000);
      });
    });
  </script>
</body>
```

3. jQuery HTML

3.1. jQuery Get Content and Attributes

One very important part of jQuery is the possibility to manipulate the DOM. jQuery comes with a bunch of DOM related methods that make it easy to access and manipulate elements and attributes. Three simple, but useful, jQuery methods for DOM manipulation are:

- *text*() - Sets or returns the text content of selected elements
- *html*() - Sets or returns the content of selected elements (including HTML markup)
- *val*() - Sets or returns the value of form fields

The following example demonstrates how to get content with the jQuery *text*() and *html*() methods:

```
<body>
  <p id="test">This is some <b>bold</b> text in a paragraph.</p>
  <button id="btn1">Show Text</button>
  <button id="btn2">Show HTML</button>
  <script src="jquery.js"></script>
  <script>
    $(document).ready(function () {
      $("#btn1").click(function () {
        alert("Text: " + $("#test").text());
      });
      $("#btn2").click(function () {
        alert("HTML: " + $("#test").html());
      });
    });
  </script>
</body>
```

```
<body>
  <p>Name: <input type="text" id="test" value="Mickey Mouse"></p>
  <button>Show Value</button>
  <script src="jquery.js"></script>
  <script>
    $(document).ready(function () {
      $("button").click(function () {
        alert("Value: " + $("#test").val());
      });
    });
  </script>
</body>
```

3.2. jQuery Set Content and Attributes

We will use the same three methods from the previous page to set content:

- *text*() - Sets or returns the text content of selected elements
- *html*() - Sets or returns the content of selected elements (including HTML markup)
- *val*() - Sets or returns the value of form fields

The following example demonstrates how to set content with the jQuery *text*(), *html*(), and *val*() methods:

```
<body>
  <p id="test1">This is a paragraph.</p>
  <p id="test2">This is another paragraph.</p>
  <p>Input field: <input type="text" id="test3" value="Mickey Mouse"></p>
  <button id="btn1">Set Text</button>
  <button id="btn2">Set HTML</button>
  <button id="btn3">Set Value</button>
  <script src="jquery.js"></script>
  <script>
    $(document).ready(function () {
      $("#btn1").click(function () {
        $("#test1").text("Hello world!");
      });
      $("#btn2").click(function () {
        $("#test2").html("<b>Hello world!</b>");
      });
      $("#btn3").click(function () {
        $("#test3").val("Hello from btn3");
      });
    });
  </script>
</body>
```

3.3. jQuery Add Elements

There are four jQuery methods that are used to add new content:

- *append*() - Inserts content at the end of the selected elements
- *prepend*() - Inserts content at the beginning of the selected elements
- *after*() - Inserts content after the selected elements
- *before*() - Inserts content before the selected elements

```
<body>
<p>This is a paragraph.</p>
<p>This is another paragraph.</p>
<ol>
  <li>List item 1</li>
  <li>List item 2</li>
  <li>List item 3</li>
</ol>
<button id="btn1">Append text</button>
<button id="btn2">Prepended list items</button>
<script src="jquery.js"></script>
<script>
$(document).ready(function(){
  $("#btn1").click(function(){
    $("p").append(" <b>Appended text</b>.");
  });
  $("#btn2").click(function(){
    $("ol").prepend("<li>Prepended item</li>");
  });
});
</script></body>
```

```
<body>
  <img src="images/cherry.jpg" alt="jQuery" width="100"
height="140"><br><br>
  <button id="btn1">Insert before</button>
  <button id="btn2">Insert after</button>
  <script src="jquery.js"></script>
  <script>
    $(document).ready(function () {
      $("#btn1").click(function () {
        $("img").before("<b>Before</b>");
      });

      $("#btn2").click(function () {
        $("img").after("<i>After</i>");
      });
    });
  </script>
</body>
```

4. Query Plug-in

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.

4.1. How to use Plug-in?

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. Following example shows how to include jquery.plug-in.js plugin −

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Calendar</title>
    <link rel="stylesheet" href="jquery-ui.css">
</head>
<body>
    <input type="text" id="date">
    <script src="jquery.js"></script>
    <script src="jquery-ui.js"></script>
    <script>
        $(document).ready(function () {
            $("#date").datepicker();
        });
    </script>
</body>
</html>
```

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Data Table</title>
    <link rel="stylesheet" href="jquery.dataTables.min.css">
</head>
<body>
    <table id="example" class="display" style="width: 100%">
        <thead>
            <tr>
                <th>Name</th>
                <th>Position</th>
                <th>Office</th>
                <th>Age</th>
                <th>Start date</th>
                <th>Salary</th>
            </tr>
        </thead>
        <tbody>
            <tr>
                <td>Tiger Nixon</td>
                <td>System Architect</td>
                <td>Edinburgh</td>
                <td>61</td>
                <td>2011/04/25</td>
                <td>$320,800</td>
            </tr>
            <tr>
                <td>Garrett Winters</td>
                <td>Accountant</td>
                <td>Tokyo</td>
                <td>63</td>
                <td>2011/07/25</td>
                <td>$170,750</td>
            </tr>
          …………………………..
        </tbody>

    </table>
    <script src="jquery.js"></script>
    <script src="jquery.dataTables.min.js"></script>
    <script>
        $(document).ready(function() {
            $('#example').DataTable();
        });
    </script>
</body>
</html>
```

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
</head>
<body>
    <form id="comment" action="">
        <p>
            <label for="cname">Name (required, at least 2 characters)</label>
            <input id="cname" type="text" name="name">
        <p>
            <label for="cemail">E-Mail (required)</label>
            <input id="cemail" type="text" name="email" />
        </p>
        <p>
            <label for="curl">URL (optional)</label>
            <input id="curl" type="text" name="url" />
        </p>
        <p>
            <label for="ccomment">Your comment (required)</label>
            <textarea id="ccomment" name="comment"></textarea>
        </p>
        <p>
            <input class="submit" type="submit" value="Submit" />
        </p>
    </form>
    <style>
        label {
            display: block;
            color: #555;
        }
        label.error {
            color: #900;
            font-style: italic;
        }
    </style>
    <script src="jquery.js"></script>
    <script src="jquery.validate.js"></script>
    <script>
        $(document).ready(function () {
            $("#comment").validate({
                rules: {
                    name: {
                        "required": true,
                        "minlength": 2
                    },
                    email: {
                        "required": true,
                        "email": true
                    },
                    comment: {
                        "required": true
                    },
                    url: {
                        "required": true,
                        "url": true
                    }
                }
            });
        });
    </script>
</body>
</html>
```

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
</head>
<body>
    <div id="wrap">
        <h1>Task List <span>0</span></h1>
        <ul id="tasks"></ul>
        <ul id="done"></ul>
        <div id="new-task">
            <input type="text"><button>+</button>
        </div>
    </div>
    <style>
        #wrap {
            width: 360px;
            margin: 20px auto;
            padding: 20px;
            border: 6px solid #DDD;
            overflow: hidden;
        }
        h1 {
            margin: 0 0 20px 0;
            padding: 0 0 5px 0;
            font-size: 18px;
            border-bottom: 1px solid #DDD;
        }
        h1 span {
            float: right;
            display: block;
            background: #9cf;
            font-size: 12px;
            padding: 2px 6px;
            color: #FFF;
        }
        ul {
            list-style: none;
            margin: 0;
            padding: 0;
        }
        ul li {
            overflow: hidden;
            padding: 4px 0;
        }
        li input,
        li span {
            float: left;
            margin-right: 6px;
        }

        li a {
            float: right;
            text-decoration: none;
            font-weight: bold;
            color: #900;
            padding: 0 8px;
            display: none
        }
        li:hover a {
            display: inline;
        }
        #new-task {
            margin-top: 20px;
            padding-top: 10px;
            border-top: 1px solid #efefef;
            overflow: hidden;
        }
        #new-task input {
            border: 1px solid #DDD;
            border-right: 0 none;
            padding: 2px;
            width: 240px;
            height: 20px;
            float: left;
        }
        #new-task button {
            width: 30px;
            border: 1px solid #DDD;
            background: #eee;
            font-weight: bold;
            color: #666;
            padding: 2px 2px 3px 2px;
            float: left;
        }
        #done {
            line-height: 18px;
            margin-bottom: 10px;
        }
        #done span {
            text-decoration: line-through;
            font-size: 12px;
            color: rgb(149, 10, 10);
        }
```

```
    </style>
    <script src="jquery.js"></script>
    <script>
        $(document).ready(function () {
            $("#new-task button").click(function () {
                var task = $("#new-task input").val();
                if (!task) return false;
                buildTask(task).appendTo("#tasks");
                $("h1 span").html($("#tasks li").length);
                $("#new-task input").val("").focus();
            });
            $("#new-task input").keydown(function (e) {
                if (e.which == 13)
                    $("#new-task button").click();
            });
        });
        function buildTask(msg) {
            var checkbox = $("<input>", {
                type: "checkbox"
            }).click(function () {
                if ($(this).is(":checked")) {
                    $(this).parent().prependTo("#done");
                } else {
                    $(this).parent().appendTo("#tasks");
                }
                $("h1 span").html($("#tasks li").length);
            });
            var task = $("<span>").html(msg);
            var del = $("<a>", {
                href: "#"
            }).html("&times;").click(function () {
                $(this).parent().remove();
                $("h1 span").html($("#tasks li").length);
            });
            return $("<li>").append(checkbox).append(task).append(del);
        }
     </script>
 </body>
 </html>
```