1. **<script>**
2. function validateform(){
3. var name=document.myform.name.value;
4. var password=document.myform.password.value;
6. if (name==null || name==""){
7. alert("Name can't be blank");
8. return false;
9. }else if(password.length**<6**){
10. alert("Password must be at least 6 characters long.");
11. return false;
12. }
13. }
14. **</script>**
15. **<body>**
16. **<form** name="myform" method="post" action="abc.jsp" onsubmit="return validateform()" **>**
17. Name: **<input** type="text" name="name"**><br/>**
18. Password: **<input** type="password" name="password"**><br/>**
19. **<input** type="submit" value="register"**>**
20. **</form>**

JavaScript Retype Password Validation

1. **<script** type="text/javascript"**>**
2. function matchpass(){
3. var firstpassword=document.f1.password.value;
4. var secondpassword=document.f1.password2.value;
6. if(firstpassword==secondpassword){
7. return true;
8. }
9. else{
10. alert("password must be same!");
11. return false;
12. }
13. }
14. **</script>**
16. **<form** name="f1" action="register.jsp" onsubmit="return matchpass()"**>**
17. Password:**<input** type="password" name="password" **/><br/>**
18. Re-enter Password:**<input** type="password" name="password2"**/><br/>**
19. **<input** type="submit"**>**
20. **</form>**

JavaScript Number Validation

Let's validate the textfield for numeric value only. Here, we are using isNaN() function.

1. **<script>**
2. function validate(){
3. var num=document.myform.num.value;
4. if (isNaN(num)){
5. document.getElementById("numloc").innerHTML="Enter Numeric value only";
6. return false;
7. }else{
8. return true;
9. }
10. }
11. **</script>**
12. **<form** name="myform" onsubmit="return validate()" **>**
13. Number: **<input** type="text" name="num"**><span** id="numloc"**></span><br/>**
14. **<input** type="submit" value="submit"**>**
15. **</form>**

JavaScript validation with image

Let’s see an interactive JavaScript form validation example that displays correct and incorrect image if input is correct or incorrect.

1. **<script>**
2. function validate(){
3. var name=document.f1.name.value;
4. var password=document.f1.password.value;
5. var status=false;
7. if(name.length**<1**){
8. document.getElementById("nameloc").innerHTML=
9. " <img src='unchecked.gif'/> Please enter your name";
10. status=false;
11. }else{
12. document.getElementById("nameloc").innerHTML=" <img src='checked.gif'/>";
13. status=true;
14. }
15. if(password.length**<6**){
16. document.getElementById("passwordloc").innerHTML=
17. " <img src='unchecked.gif'/> Password must be at least 6 char long";
18. status=false;
19. }else{
20. document.getElementById("passwordloc").innerHTML=" <img src='checked.gif'/>";
21. }
22. return status;
23. }
24. **</script>**
26. **<form** name="f1" action="#" onsubmit="return validate()"**>**
27. **<table>**
28. **<tr><td>**Enter Name:**</td><td><input** type="text" name="name"**/>**
29. **<span** id="nameloc"**></span></td></tr>**
30. **<tr><td>**Enter Password:**</td><td><input** type="password" name="password"**/>**
31. **<span** id="passwordloc"**></span></td></tr>**
32. **<tr><td** colspan="2"**><input** type="submit" value="register"**/></td></tr>**
33. **</table>**
34. **</form>**

JavaScript email validation

We can validate the email by the help of JavaScript.

There are many criteria that need to be follow to validate the email id such as:

* email id must contain the @ and . character
* There must be at least one character before and after the @.
* There must be at least two characters after . (dot).

Let's see the simple example to validate the email field.

1. **<script>**
2. function validateemail()
3. {
4. var x=document.myform.email.value;
5. var atposition=x.indexOf("@");
6. var dotposition=x.lastIndexOf(".");
7. if (atposition**<1** || dotposition**<atposition**+2 || dotposition+2**>**=x.length){
8. alert("Please enter a valid e-mail address \n atpostion:"+atposition+"\n dotposition:"+dotposition);
9. return false;
10. }
11. }
12. **</script>**
13. **<body>**
14. **<form** name="myform"  method="post" action="#" onsubmit="return validateemail();"**>**
15. Email: **<input** type="text" name="email"**><br/>**
17. **<input** type="submit" value="register"**>**
18. **</form>**
19. document.cookie="name=value";

## JavaScript Cookie Example

### **Example 1**

Let's see an example to set and get a cookie.

1. <!DOCTYPE html**>**
2. **<html>**
3. **<head>**
4. **</head>**
5. **<body>**
6. **<input** type="button" value="setCookie" onclick="setCookie()"**>**
7. **<input** type="button" value="getCookie" onclick="getCookie()"**>**
8. **<script>**
9. function setCookie()
10. {
11. document.cookie="username=Duke Martin";
12. }
13. function getCookie()
14. {
15. if(document.cookie.length!=0)
16. {
17. alert(document.cookie);
18. }
19. else
20. {
21. alert("Cookie not available");
22. }
23. }
24. **</script>**
26. **</body>**
27. **</html>**

### **Example 2**

Here, we display the cookie's name-value pair separately.

1. <!DOCTYPE html**>**
2. **<html>**
3. **<head>**
4. **</head>**
5. **<body>**
6. **<input** type="button" value="setCookie" onclick="setCookie()"**>**
7. **<input** type="button" value="getCookie" onclick="getCookie()"**>**
8. **<script>**
9. function setCookie()
10. {
11. document.cookie="username=Duke Martin";
12. }
13. function getCookie()
14. {
15. if(document.cookie.length!=0)
16. {
17. var array=document.cookie.split("=");
18. alert("Name="+array[0]+" "+"Value="+array[1]);
19. }
20. else
21. {
22. alert("Cookie not available");
23. }
24. }
25. **</script>**
27. **</body>**
28. **</html>**

### **Example 3**

In this example, we provide choices of color and pass the selected color value to the cookie. Now, cookie stores the last choice of a user in a browser. So, on reloading the web page, the user's last choice will be shown on the screen.

1. <!DOCTYPE html**>**
2. **<html>**
3. **<head>**
4. **</head>**
5. **<body>**
6. **<select** id="color" onchange="display()"**>**
7. **<option** value="Select Color"**>**Select Color**</option>**
8. **<option** value="yellow"**>**Yellow**</option>**
9. **<option** value="green"**>**Green**</option>**
10. **<option** value="red"**>**Red**</option>**
11. **</select>**
12. **<script** type="text/javascript"**>**
13. function display()
14. {
15. var value = document.getElementById("color").value;
16. if (value != "Select Color")
17. {
18. document.bgColor = value;
19. document.cookie = "color=" + value;
20. }
21. }
22. window.onload = function ()
23. {
24. if (document.cookie.length != 0)
25. {
26. var array = document.cookie.split("=");
27. document.getElementById("color").value = array[1];
28. document.bgColor = array[1];
29. }
30. }

33. **</script>**
34. **</body>**
35. **</html>**

Cookie expires attribute

The cookie expires attribute provides one of the ways to create a persistent cookie. Here, a date and time are declared that represents the active period of a cookie. Once the declared time is passed, a cookie is deleted automatically.

Let's see an example of cookie expires attribute.

1. <!DOCTYPE html**>**
2. **<html>**
3. **<head>**
4. **</head>**
5. **<body>**
6. **<input** type="button" value="setCookie" onclick="setCookie()"**>**
7. **<input** type="button" value="getCookie" onclick="getCookie()"**>**
8. **<script>**
9. function setCookie()
10. {
11. document.cookie="username=Duke Martin;expires=Sun, 20 Aug 2030 12:00:00 UTC";
12. }
13. function getCookie()
14. {
15. if(document.cookie.length!=0)
16. {
17. var array=document.cookie.split("=");
18. alert("Name="+array[0]+" "+"Value="+array[1]);
19. }
20. else
21. {
22. alert("Cookie not available");
23. }
24. }
25. **</script>**
26. **</body>**
27. **</html>**

Cookie max-age attribute

The cookie max-age attribute provides another way to create a persistent cookie. Here, time is declared in seconds. A cookie is valid up to the declared time only.

Let's see an example of cookie max-age attribute.

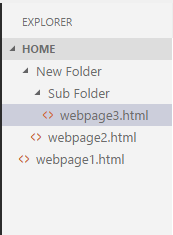
1. <!DOCTYPE html**>**
2. **<html>**
3. **<head>**
4. **</head>**
5. **<body>**
6. **<input** type="button" value="setCookie" onclick="setCookie()"**>**
7. **<input** type="button" value="getCookie" onclick="getCookie()"**>**
8. **<script>**
9. function setCookie()
10. {
11. document.cookie="username=Duke Martin;max-age=" + (60 \* 60 \* 24 \* 365) + ";"
12. }
13. function getCookie()
14. {
15. if(document.cookie.length!=0)
16. {
17. var array=document.cookie.split("=");
18. alert("Name="+array[0]+" "+"Value="+array[1]);
19. }
20. else
21. {
22. alert("Cookie not available");
23. }
24. }
25. **</script>**
26. **</body>**
27. **</html>**

Cookie path attribute

If a cookie is created for a webpage, by default, it is valid only for the current directory and sub-directory. JavaScript provides a path attribute to expand the scope of cookie up to all the pages of a website.

Cookie path attribute Example

Let's understand the path attribute with the help of an example.



Here, if we create a cookie for webpage2.html, it is valid only for itself and its sub-directory (i.e., webpage3.html). It is not valid for webpage1.html file.

In this example, we use path attribute to enhance the visibility of cookies up to all the pages. Here, you all just need to do is to maintain the above directory structure and put the below program in all three web pages. Now, the cookie is valid for each web page.

1. <!DOCTYPE html**>**
2. **<html>**
3. **<head>**
4. **</head>**
5. **<body>**
6. **<input** type="button" value="setCookie" onclick="setCookie()"**>**
7. **<input** type="button" value="getCookie" onclick="getCookie()"**>**
8. **<script>**
9. function setCookie()
10. {
11. document.cookie="username=Duke Martin;max-age=" + (60 \* 60 \* 24 \* 365) + ";path=/;"
12. }
13. function getCookie()
14. {
15. if(document.cookie.length!=0)
16. {
17. var array=document.cookie.split("=");
18. alert("Name="+array[0]+" "+"Value="+array[1]);
19. }
20. else
21. {
22. alert("Cookie not available");
23. }
24. }
25. **</script>**
26. **</body>**
27. **</html>**

Cookie domain attribute

A JavaScript domain attribute specifies the domain for which the cookie is valid. Let's suppose if we provide any domain name to the attribute such like:

1. domain=javatpoint.com

Here, the cookie is valid for the given domain and all its sub-domains.

However, if we provide any sub-domain to the attribute such like:

1. omain=training.javatpoint.com

Here, the cookie is valid only for the given sub-domain. So, it's a better approach to provide domain name instead of sub-domain.

# **Cookie with multiple Name-Value pairs**

In JavaScript, a cookie can contain only a single name-value pair. However, to store more than one name-value pair, we can use the following approach: -

* Serialize the custom object in a JSON string, parse it and then store in a cookie.
* For each name-value pair, use a separate cookie.

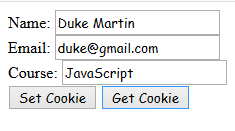
## Examples to Store Name-Value pair in a Cookie

### **Example 1**

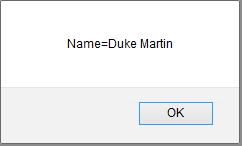
Let's see an example to check whether a cookie contains more than one name-value pair.

1. <!DOCTYPE html**>**
2. **<html>**
3. **<head>**
4. **</head>**
5. **<body>**
6. Name: **<input** type="text" id="name"**><br>**
7. Email: **<input** type="email" id="email"**><br>**
8. Course: **<input** type="text" id="course"**><br>**
9. **<input** type="button" value="Set Cookie" onclick="setCookie()"**>**
10. **<input** type="button" value="Get Cookie" onclick="getCookie()"**>**
11. **<script>**
12. function setCookie()
13. {
14. //Declaring 3 key-value pairs
15. var info="Name="+ document.getElementById("name").value+";Email="+document.getElementById("email").value+";Course="+document.getElementById("course").value;
16. //Providing all 3 key-value pairs to a single cookie
17. document.cookie=info;
18. }
20. function getCookie()
21. {
22. if(document.cookie.length!=0)
23. {
24. //Invoking key-value pair stored in a cookie
25. alert(document.cookie);
26. }
27. else
28. {
29. alert("Cookie not available")
30. }
31. }
32. **</script>**
33. **</body>**
34. **</html>**

**Output:**

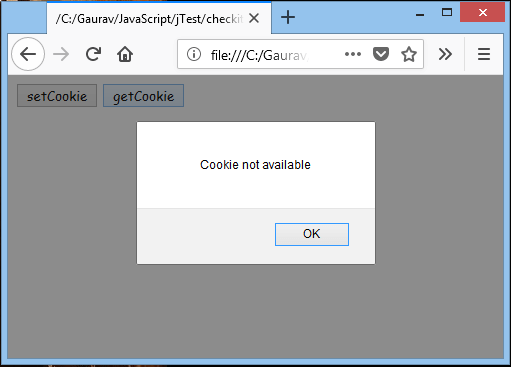
[**Test it Now**](https://www.javatpoint.com/oprweb/test.jsp?filename=JavaScriptCookiewithmultipleNameExample1) 

On clicking **Get Cookie** button, the below dialog box appears.



Here, we can see that only a single name-value is displayed.

However, if you click, **Get Cookie** without filling the form, the below dialog box appears.



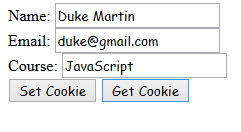
### **Example 2**

Let's see an example to store different name-value pairs in a cookie using JSON.

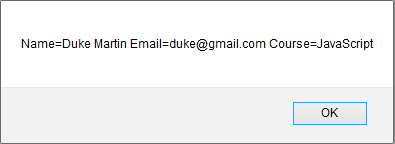
1. <!DOCTYPE html**>**
2. **<html>**
3. **<head>**
4. **</head>**
5. **<body>**
6. Name: **<input** type="text" id="name"**><br>**
7. Email: **<input** type="email" id="email"**><br>**
8. Course: **<input** type="text" id="course"**><br>**
9. **<input** type="button" value="Set Cookie" onclick="setCookie()"**>**
10. **<input** type="button" value="Get Cookie" onclick="getCookie()"**>**
12. **<script>**
13. function setCookie()
14. {
15. var obj = {};//Creating custom object
16. obj.name = document.getElementById("name").value;
17. obj.email = document.getElementById("email").value;
18. obj.course = document.getElementById("course").value;
20. //Converting JavaScript object to JSON string
21. var jsonString = JSON.stringify(obj);
23. document.cookie = jsonString;
24. }
25. function getCookie()
26. {
27. if( document.cookie.length!=0)
28. {
29. //Parsing JSON string to JSON object
30. var obj = JSON.parse(document.cookie);
32. alert("Name="+obj.name+" "+"Email="+obj.email+" "+"Course="+obj.course);
33. }
34. else
35. {
36. alert("Cookie not available");
37. }
38. }
39. **</script>**
40. **</body>**
41. **</html>**

[**Test it Now**](https://www.javatpoint.com/oprweb/test.jsp?filename=JavaScriptCookiewithmultipleNameExample2)

**Output:**



On clicking **Get Cookie** button, the below dialog box appears.



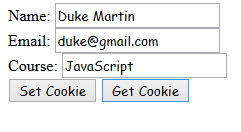
Here, we can see that all the stored name-value pairs are displayed.

### **Example 3**

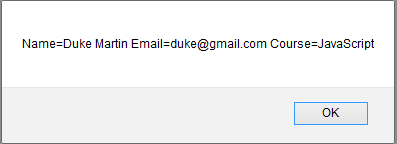
Let's see an example to store each name-value pair in a different cookie.

1. <!DOCTYPE html**>**
2. **<html>**
3. **<head>**
4. **</head>**
5. **<body>**
6. Name: **<input** type="text" id="name"**><br>**
7. Email: **<input** type="email" id="email"**><br>**
8. Course: **<input** type="text" id="course"**><br>**
9. **<input** type="button" value="Set Cookie" onclick="setCookie()"**>**
10. **<input** type="button" value="Get Cookie" onclick="getCookie()"**>**
12. **<script>**
13. function setCookie()
14. {
15. document.cookie = "name=" + document.getElementById("name").value;
16. document.cookie = "email=" + document.getElementById("email").value;
17. document.cookie = "course=" + document.getElementById("course").value;
18. }
19. function getCookie()
20. {
21. if (document.cookie.length != 0)
22. {
23. alert("Name="+document.getElementById("name").value+" Email="+document.getElementById("email").value+" Course="+document.getElementById("course").value);
24. }
25. else
26. {
27. alert("Cookie not available");
28. }
29. }
30. **</script>**
31. **</body>**
32. **</html>**

**Output:**

[**Test it Now**](https://www.javatpoint.com/oprweb/test.jsp?filename=JavaScriptCookiewithmultipleNameExample3) 

On clicking **Get Cookie** button, the below dialog box appears.



Here, also we can see that all the stored name-value pairs are displayed.

# **Deleting a Cookie in JavaScript**

In the previous section, we learned the different ways to set and update a cookie in JavaScript. Apart from that, JavaScript also allows us to delete a cookie. Here, we see all the possible ways to delete a cookie.

## Different ways to delete a Cookie

These are the following ways to delete a cookie:

* A cookie can be deleted by using expire attribute.
* A cookie can also be deleted by using max-age attribute.
* We can delete a cookie explicitly, by using a web browser.

## Examples to delete a Cookie

### **Example 1**

In this example, we use expire attribute to delete a cookie by providing expiry date (i.e. any past date) to it.

1. <!DOCTYPE html**>**
2. **<html>**
3. **<head>**
4. **</head>**
5. **<body>**
7. **<input** type="button" value="Set Cookie" onclick="setCookie()"**>**
8. **<input** type="button" value="Get Cookie" onclick="getCookie()"**>**
9. **<script>**
10. function setCookie()
11. {
12. document.cookie="name=Martin Roy; expires=Sun, 20 Aug 2000 12:00:00 UTC";
14. }
15. function getCookie()
16. {
17. if(document.cookie.length!=0)
18. {
19. alert(document.cookie);
20. }
21. else
22. {
23. alert("Cookie not avaliable");
24. }
25. }
26. **</script>**
27. **</body>**
28. **</html>**

### **Example 2**

In this example, we use **max-age** attribute to delete a cookie by providing zero or negative number (that represents seconds) to it.

[](https://campaign.adpushup.com/get-started/?utm_source=banner&utm_campaign=growth_hack)

1. <!DOCTYPE html**>**
2. **<html>**
3. **<head>**
4. **</head>**
5. **<body>**
7. **<input** type="button" value="Set Cookie" onclick="setCookie()"**>**
8. **<input** type="button" value="Get Cookie" onclick="getCookie()"**>**
9. **<script>**
10. function setCookie()
11. {
12. document.cookie="name=Martin Roy;max-age=0";
13. }
14. function getCookie()
15. {
16. if(document.cookie.length!=0)
17. {
18. alert(document.cookie);
19. }
20. else
21. {
22. alert("Cookie not avaliable");
23. }
24. }
26. **</script>**
27. **</body>**
28. **</html>**

### **Example 3**

Let's see an example to set, get and delete multiple cookies.

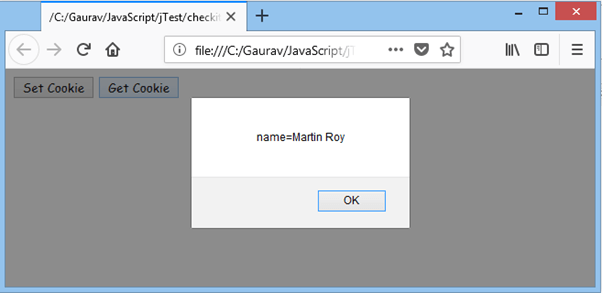
1. <!DOCTYPE html**>**
2. **<html>**
3. **<head>**
4. **</head>**
5. **<body>**
7. **<input** type="button" value="Set Cookie1" onclick="setCookie1()"**>**
8. **<input** type="button" value="Get Cookie1" onclick="getCookie1()"**>**
9. **<input** type="button" value="Delete Cookie1" onclick="deleteCookie1()"**>**
10. **<br>**
11. **<input** type="button" value="Set Cookie2" onclick="setCookie2()"**>**
12. **<input** type="button" value="Get Cookie2" onclick="getCookie2()"**>**
13. **<input** type="button" value="Delete Cookie2" onclick="deleteCookie2()"**>**
14. **<br>**
15. **<input** type="button" value="Display all cookies" onclick="displayCookie()"**>**
17. **<script>**
18. function setCookie1()
19. {
20. document.cookie="name=Martin Roy";
21. cookie1=  document.cookie;
22. }
23. function setCookie2()
24. {
25. document.cookie="name=Duke William";
26. cookie2=  document.cookie;
27. }
29. function getCookie1()
30. {
31. if(cookie1.length!=0)
32. {
33. alert(cookie1);
34. }
35. else
36. {
37. alert("Cookie not available");
38. }
39. }
41. function getCookie2()
42. {
43. if(cookie2.length!=0)
44. {
45. alert(cookie2);
46. }
47. else
48. {
49. alert("Cookie not available");
50. }
51. }
53. function deleteCookie1()
54. {
55. document.cookie=cookie1+";max-age=0";
56. cookie1=document.cookie;
57. alert("Cookie1 is deleted");
58. }
60. function deleteCookie2()
61. {
62. document.cookie=cookie2+";max-age=0";
63. cookie2=document.cookie;
64. alert("Cookie2 is deleted");
65. }
67. function displayCookie()
68. {
69. if(cookie1!=0&&cookie2!=0)
70. {
71. alert(cookie1+" "+cookie2);
72. }
73. else if(cookie1!=0)
74. {
75. alert(cookie1);
76. }
77. else if(cookie2!=0)
78. {
79. alert(cookie2);
80. }
81. else{
82. alert("Cookie not available");
83. }
85. }
87. **</script>**
88. **</body>**
89. **</html>**

### **Example 4**

Let's see an example to delete a cookie explicitly.

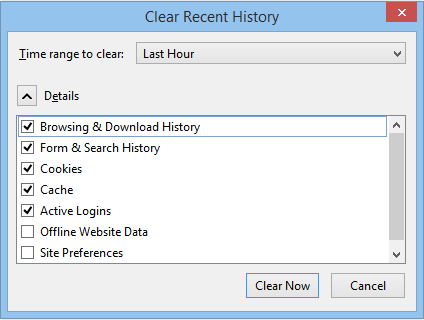
1. <!DOCTYPE html**>**
2. **<html>**
3. **<head>**
4. **</head>**
5. **<body>**
7. **<input** type="button" value="Set Cookie" onclick="setCookie()"**>**
8. **<input** type="button" value="Get Cookie" onclick="getCookie()"**>**
9. **<script>**
10. function setCookie()
11. {
12. document.cookie="name=Martin Roy";
14. }
15. function getCookie()
16. {
17. if(document.cookie.length!=0)
18. {
19. alert(document.cookie);
20. }
21. else
22. {
23. alert("Cookie not avaliable");
24. }
25. }
26. **</script>**
27. **</body>**
28. **</html>**

After clicking **Set Cookie** once, whenever we click **Get Cookie**, the cookies key and value is displayed on the screen.



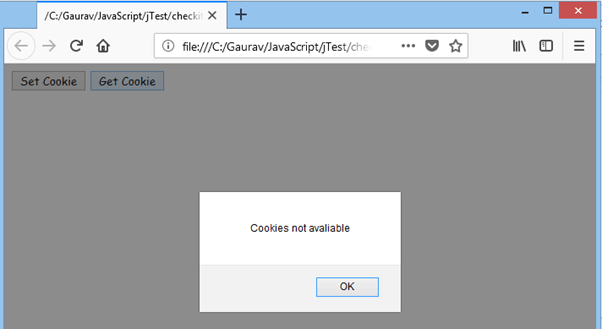
To delete a cookie explicitly, follow the following steps:

* Open Mozilla Firefox.
* Click **Open menu - Library - History - Clear Recent History - Details**.



* Here we can see a **Cookies** checkbox which is already marked. Now, click **Clear Now** to delete the cookies explicitly.

Now, on clicking **Get Cookie**, the below dialog box appears.



Here, we can see that the cookies are deleted.