Лекция №3
Cookie, DOM и События

План:

1. Cookie
2. DOM и События

Файл cookie — это фрагмент данных, который хранится на вашем компьютере для доступа к вашему браузеру. Вы также могли бы использовать преимущества куки сознательно или неосознанно. Вы когда-нибудь сохраняли свой пароль Facebook, чтобы вам не приходилось вводить его каждый раз, когда вы пытаетесь войти? Если да, то вы используете куки. Файлы cookie сохраняются в виде пар ключ / значение.

Зачем вам нужен Cookie?

Связь между веб-браузером и сервером происходит по протоколу без сохранения состояния с именем HTTP. Протокол без сохранения состояния обрабатывает каждый запрос независимо. Таким образом, сервер не сохраняет данные после отправки их в браузер. Но во многих ситуациях данные снова потребуются. Вот и печеньки в картинке. При использовании файлов cookie веб-браузеру не нужно будет обмениваться данными с сервером каждый раз, когда требуются данные. Вместо этого его можно получить непосредственно с компьютера.

Javascript Set Cookie

Вы можете создавать куки, используя документ. свойство cookie, как это.

```
document.cookie = "cookiename=cookievalue"
```

Вы даже можете добавить дату истечения срока действия в ваш файл cookie, чтобы конкретный файл cookie был удален с компьютера в указанную дату. Дата истечения должна быть установлена в формате UTC / GMT. Если вы не установите дату истечения срока действия, cookie будет удален, когда пользователь закроет браузер.

```
document.cookie = "cookiename=cookievalue; expires= Thu, 21 Aug 2014 20:00:00 UTC"
```

Вы также можете указать домен и путь, чтобы указать, какому домену и каким каталогам в конкретном домене принадлежит cookie. По умолчанию cookie принадлежит странице, которая устанавливает cookie.

```
document.cookie = "cookiename=cookievalue; expires= Thu, 21 Aug 2014 20:00:00 UTC; path=/ "
```

// создаем куки с доменом для текущей страницы и путем к всему домену.

JavaScript получить Cookie

Вы можете получить доступ к таким файлам cookie, которые будут возвращать все файлы cookie, сохраненные для текущего домена.

```
var x =  document.cookie
```

JavaScript Удалить Cookie

Чтобы удалить cookie, вам просто нужно установить значение cookie пустым и установить значение expires на прошедшую дату.

```
document.cookie = "cookiename= ; expires = Thu, 01 Jan 1970 00:00:00 GMT"
```

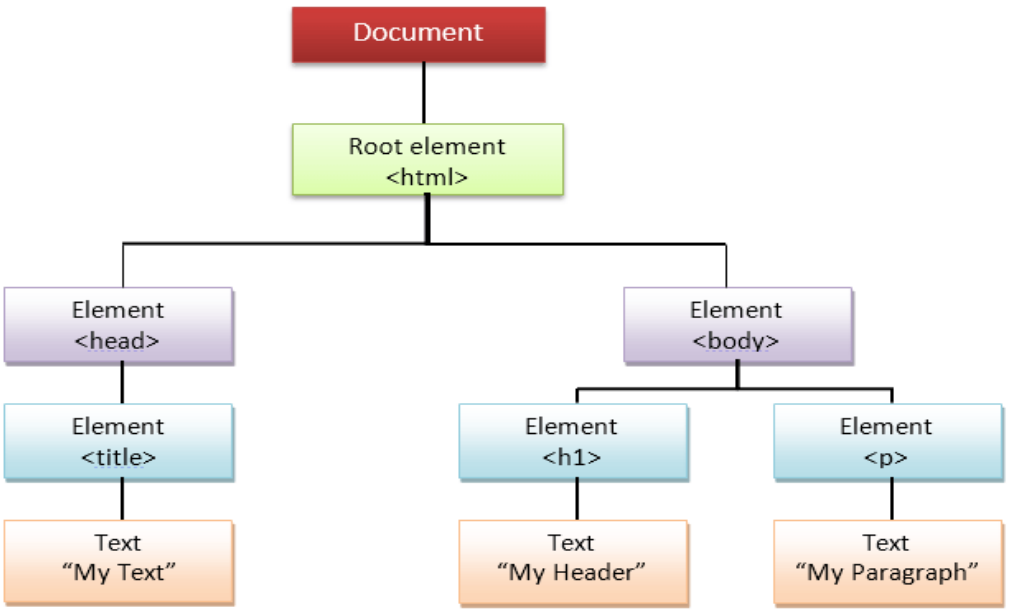
Попробуйте этот пример самостоятельно:

```
<html>
<head>
  <title>Cookie!!!</title>
```

```
<script type="text/javascript">
  function createCookie(cookieName,cookieValue,daysToExpire)
  {
   var date = new Date();
   date.setTime(date.getTime()+(daysToExpire*24*60*60*1000));
                document.cookie = cookieName + "=" + cookieValue + "; expires=" +
date.toGMTString();
  }
  function accessCookie(cookieName)
  {
   var name = cookieName + "=";
   var allCookieArray = document.cookie.split(';');
   for(var i=0; i<allCookieArray.length; i++)
   {
    var temp = allCookieArray[i].trim();
    if (temp.indexOf(name)==0)
    return temp.substring(name.length,temp.length);
   }
    return "";
  }
  function checkCookie()
  {
   var user = accessCookie("testCookie");
   if (user!="")
    alert("Welcome Back " + user + "!!!");
   else
   {
    user = prompt("Please enter your name");
```

DOM и События

Что такое DOM в JavaScript? JavaScript может получить доступ ко всем элементам веб-страницы, используя объектную модель документа (DOM). Фактически, веб-браузер создает DOM веб-страницы при загрузке страницы. Модель DOM создается в виде дерева таких объектов:



DOM и События

Используя DOM, JavaScript может выполнять несколько задач. Он может создавать новые элементы и атрибуты, изменять существующие элементы и атрибуты и даже удалять существующие элементы и атрибуты. JavaScript также может реагировать на существующие события и создавать новые события на странице.

getElementById, innerHTML Пример

1. getElementById: для доступа к элементам и атрибутам, чей идентификатор установлен.
2. innerHTML: для доступа к содержимому элемента.

Попробуйте этот пример самостоятельно:

```
<html>
<head>
   <title>DOM!!!</title>
</head>
<body>
  <h1 id="one">Welcome</h1>
  <p>This is the welcome message.</p>
  <h2>Technology</h2>
  <p>This is the technology section.</p>
  <script type="text/javascript">
       var text = document.getElementById("one").innerHTML;
       alert("The first heading is " + text);
  </script>
</body>
</html>
```

Пример getElementsByTagName

getElementsByTagName: для доступа к элементам и атрибутам, используя имя тега. Этот метод возвращает массив всех элементов с одинаковым именем тега.

Попробуйте этот пример самостоятельно:

```
<html>
<head>
   <title>DOM!!!</title>
</head>
<body>
  <h1>Welcome</h1>
  <p>This is the welcome message.</p>
  <h2>Technology</h2>
  <p id="second">This is the technology section.</p>
  <script type="text/javascript">
    var paragraphs = document.getElementsByTagName("p");
    alert("Content in the second paragraph is " + paragraphs[1].innerHTML);
    document.getElementById("second").innerHTML = "The orginal message is changed.";
  </script>
</body>
</html>
```

Пример обработчика событий

1. createElement: создать новый элемент
2. removeChild: удалить элемент

3. Вы можете добавить **обработчик события** к определенному элементу, например так:

```
        document.getElementById(id).onclick=function()
    {
        lines of code to be executed
    }
```

ИЛИ

```
document.getElementById(id).addEventListener("click", functionname)
```

Попробуйте этот пример самостоятельно:

```
<html>
<head>
   <title>DOM!!!</title>
</head>
<body>
  <input type="button" id="btnClick" value="Click Me!!" />
  <script type="text/javascript">
    document.getElementById("btnClick").addEventListener("click", clicked);
    function clicked()
    {
        alert("You clicked me!!!");
    }
  </script>
</body>
</html>
```