Cookies и механизм сохранения данныхна стороне клиента

В объекте Document имеется не обсуждавшееся ранее свойство под названием cookie. На первый взгляд кажется, что это свойство представляет собой простое строковое значение; однако свойство cookie – это много больше, чем просто строка: оно позволяет JavaScript коду сохранять данные на жестком диске в пользовательской системе и извлекать сохраненные ранее данные. Cookies представляют собой простой способ наделить веб-приложения долговременной памятью, например, веб-сайт может сохранять личные предпочтения пользователя и затем использовать их при повторном посещении страницы. Кроме того, cookies могут использоваться сценариями на стороне сервера и являются стандартным расширением протокола HTTP. Все современные броузеры поддерживают cookies и предоставляют к ним доступ через свойство Document.cookie. Также существует еще один более мощный механизм сохранения данных на стороне клиента, но он плохо стандартизован.

Обзор cookies

Cookie – это небольшой объем именованных данных, сохраняемых веб-броузером и связанных с определенной веб-страницей или веб-сайтом. Cookies играют роль памяти веб-броузера, чтобы сценарии и программы на стороне сервера могли на одной странице работать с данными, введенными на другой странице, или чтобы броузер мог вспомнить пользовательские параметры или другие переменные состояния, когда возвращается на страницу, посещенную им ранее.

Cookies первоначально предназначались для разработки серверных сценариев и на низшем уровне реализованы как расширение протокола HTTP. Данные cookie автоматически передаются между веб-броузером и веб-сервером, так что серверные сценарии могут читать и записывать значения cookie, сохраняемые на стороне клиента. Как мы увидим, JavaScript также может работать с cookie с помощью свойства cookie объекта Document. Свойство cookie – это строковое свойство, позволяющее читать, создавать, изменять и удалять cookies, связанные с текущей веб-страницей. Хотя cookie с первого взгляда может показаться обычным доступным для чтения и записи строковым свойством, фактически его поведение сложнее. Читая значение свойства cookie, мы получаем строку, содержащую имена и значения всех cookies, связанных с документом. Можно создавать, изменять и удалять cookies, устанавливая значение свойства cookie.

Однако для того чтобы работа со свойством cookie была эффективной, надо больше знать о cookies и о том, как они работают. Помимо имени и значения

каждый cookie имеет четыре необязательных атрибута, управляющих временем его жизни, видимостью и безопасностью.

По умолчанию cookies являются временными – их значения сохраняются на период сеанса веб-броузера и теряются при закрытии сеанса пользователем. Чтобы cookie сохранялся после окончания сеанса, необходимо сообщить броузеру, как долго он должен храниться. Изначально для этого использовался атрибут expires, указывающий дату окончания действия cookie. И хотя этот атрибут по-прежнему может применяться, он начинает вытесняться другим атрибутом – max-age, который определяет срок хранения cookie в секундах. Установка значения любого из этих атрибутов заставляет броузер сохранить cookie в локальном файле, чтобы он мог быть прочитан при следующем посещении пользователем веб-страницы. После того как будет достигнута дата окончания действия или истечет период max-age, броузер автоматически удалит cookie-файл.

Еще один немаловажный атрибут cookie, path, задает веб-страницы, с которыми связан cookie. По умолчанию cookie связывается с создавшей его веб-страницей и доступен этой же странице, а также любой другой странице из того же каталога или любых его подкаталогов. Если, например, вебстраница http://www.examp' le.com/catalog/index.html создает cookie, то этот cookie будет также видим страницам http://www.example.com/catalog/order.html и http://www.example.com/cata' log/widgets/index.html, но не видим странице http://www.example.com/about.html. Этого правила видимости, принятого по умолчанию, обычно вполне достаточно. Тем не менее иногда значения cookie-файла требуется использовать на всем многостраничном вебсайте независимо от того, какая страница создала cookie.

Например, если пользователь ввел свой адрес в форму на одной странице, целесообразно сохранить этот адрес как адрес, применяемый по умолчанию. Тогда этим адресом можно будет воспользоваться при следующем посещении тем же пользователем этой страницы, а также при заполнении им совершенно другой формы на любой другой странице, где требуется ввести адрес, например для выставления счета. Чтобы это можно было сделать, для cookie-файла задается значение path. Тогда любая страница того же веб-сервера, содержащая указанное значение в своем URL, сможет использовать cookie-файл. Например, если для cookie, установленного страницей http://www.example.com/catalog/widgets/index.html, для атрибута path задано значение "/catalog", этот cookie будет также виден для страницы http://www.example.com/catalog/order.html. А если атрибут path установлен в "/", то cookieфайл будет виден для любой страницы на вебсервере www.example.com

Cookies пользуются дурной славой у многих пользователей Всемирной паутины, поскольку сторонние производители часто недобросовестно применяют cookies, связанные не с самой вебстраницей, а с изображениями на ней. Например, cookies сторонних производителей позволяют компаниям, предоставляющим услуги рекламного характера, отслеживать перемещение пользователей с одного сайта на другой, что вынуждает многих пользователей по соображениям безопасности отключать режим сохранения cookies в своих веб-броузерах. Поэтому, прежде чем использовать cookie в сценариях JavaScript, следует проверить, не отключен ли режим их сохранения. В большинстве броузеров это можно сделать, проверив свойство navigator.cookieEnabled. Если оно содержит значение true, значит работа с cookie разрешена, а если false – запрещена (хотя при этом могут быть разрешены временные cookieфайлы, срок жизни которых ограничивается продолжительностью сеанса работы броузера). Это свойство не является стандартным, поэтому если сценарий вдруг обнаружит, что оно не определено, придется проверить, поддерживаются ли cookies, попытавшись записать, прочитать и удалить тестовый cookie-файл.

Сохранение cookie

Чтобы связать временное значение cookie-файла с текущим документом, достаточно установить свойство cookie равным строке следующего формата:

имя=значение

Например:

document.cookie = "version=" + encodeURIComponent(document.lastModified);

При следующем чтении свойства cookie сохраненная пара «имя–значение» будет включена в список cookie-файлов документа. Значения cookie не могут содержать точки с запятой, запятые или символы-разделители. По этой причине для кодирования значения перед сохранением его в cookie-файле, возможно, потребуется использовать JavaScript-функцию encodeURIComponent().

В этом случае при чтении значения cookie-файла надо будет вызвать соответствующую функцию decodeURIComponent(). (Нередко можно встретить программный код, использующий для тех же целей устаревшие функции escape() и unescape().) Записанный указанным способом cookie сохраняется в текущем сеансе работы вебброузера, но теряется при закрытии броузера пользователем. Чтобы создать cookie, сохраняющийся между сеансами броузера, необходимо указать время жизни (в секундах) с помощью

атрибута max-age. Это можно сделать, установив значение свойства cookie равным строке следующего формата:

name=value; maxage=seconds

Например, чтобы создать cookie, сохраняющийся в течение года, можно использовать следующий фрагмент:

document.cookie = "version=" + document.lastModified + "; maxage=" + (60*60*24*365);

Кроме того, существует возможность указать время жизни cookie с помощью устаревшего атрибута expires, в который необходимо записать дату в формате, возвращаемом функцией Date.toGMTString(). Например:

var nextyear = new Date(); nextyear.setFullYear(nextyear.getFullYear() + 1);

document.cookie = "version=" + document.lastModified + "; expires=" + nextyear.toGMTString();

Аналогичным образом можно установить атрибуты path, domain и secure, дописав к значению cookieфайла строки следующего формата перед его записью в свойство cookie:

; path=путь ;

domain=домен ;

secure

Чтобы изменить значение cookie, установите его значение снова, указав то же имя и новое значение. При изменении значения cookie-файла можно также переопределить время жизни, -указав новые значения для атрибута max-age или expires.

Чтобы удалить cookie, установите произвольное (возможно пустое) значение с тем же именем, а в атрибут max-age запишите 0 (или в атрибут expires запишите уже прошедшую дату). Обратите внимание: броузер не обязан удалять cookie не медленно, так что он может сохраниться броузером и после даты окончания его действия.

Ограничения cookie

Cookie-файлы рассчитаны на то, чтобы изредка сохранять небольшие объемы данных. Они не являются универсальным средством взаимодействия или механизмом передачи данных, поэтому следует проявлять умеренность при их использовании. Спецификации RFC 2965 рекомендуют производителям броузеров не ограничивать число и размеры сохраняемых cookieфайлов.

Однако следует знать, что стандарты не требуют, чтобы вебброузеры сохраняли более 300 cookies и 20 cookies на один веб-сервер (на весь вебсервер, а не только на вашу страницу или сайт на сервере) или по 4 Кбайт данных на один cookie (в этом ограничении учитываются и значение cookieфайла и его имя). На практике современные броузеры позволяют сохранять гораздо больше 300 cookies, но ограничение на размер 4 Кбайт для одного cookie в некоторых броузерах попрежнему соблюдается.

Чтение cookies

Когда свойство cookie используется в JavaScript-выражении, возвращаемое им значение содержит все cookieфайлы, относящиеся к текущему документу. Эта строка представляет собой список пар имя–значение, разделенных точками с запятой, где имя – это имя cookie-файла, а значение – его строковое значение. Это значение не включает каких-либо атрибутов, которые могли быть установлены для cookie. Для получения значения cookie с определенным именем могут использоваться методы String.indexOf() и String.substring(), а для разбиения строки cookieфайла на отдельные составляющие – метод String.split(). После извлечения значений cookieфайла из свойства cookie их требуется интерпретировать, основываясь на том формате или кодировке, которые были указаны создателем cookie. Например, в одном cookie может храниться несколько единиц информации в полях, разделенных двоеточиями. В этом случае придется для извлечения различных фрагментов информации обратиться к соответствующим строковым методам. Не забудьте вызвать функцию decodeURIComponent() для значения cookie, если оно было закодировано функцией encodeURIComponent(). Следующий фрагмент демонстрирует, как выполняется чтение свойства cookie, как из него извлекается отдельное значение и как потом можно использовать это значение:

```
// Прочитать свойство cookie. В результате будут получены все cookies данного документа.

var allcookies = document.cookie;

// Отыскать начало cookieфайла с именем "version"

var pos = allcookies.indexOf("version=");

// Если cookie с данным именем найден, извлечь и использовать его значение
if (pos != 1) {

var start = pos + 8;              // Начало значения cookie

var end = allcookies.indexOf(";", start);    // Конец значения cookie
```

```
if (end == 1) end = allcookies.length;

var value = allcookies.substring(start, end); // Извлекаем значение

value = decodeURIComponent (value);          // Декодируем его

   // Теперь, получив значение cookieфайла, мы можем его использовать.    //
В данном случае значение было установлено равным дате изменения    //
документа, поэтому мы можем использовать это значение, чтобы узнать,    //
был ли документ изменен с момента последнего посещения пользователем.
if (value != document.lastModified)

alert("Документ был изменен с момента вашего последнего посещения"); }
```

Обратите внимание: строка, полученная при чтении значения свойства cookie, не содержит какой-либо информации о различных атрибутах cookieфайла. Свойство cookie позволяет установить эти атрибуты, но не дает возможности прочитать их.

Альтернативы cookies

Cookies имеют пару недостатков, которые осложняют их использование для хранения данных на стороне клиента: • Размер cookie ограничен значением 4 Кбайт. • Даже когда cookies используются исключительно для нужд клиентских сценариев, они все равно загружаются на вебсервер при запросе любых страниц, с которыми они связаны. Если cookies не используются на сервере, они понапрасну расходуют пропускную способность. Есть две альтернативы использованию cookies. В Internet Explorer и в подключаемом Flash-модуле имеются фирменные (соответственно от Microsoft и Adobe) механизмы сохранения данных на стороне клиента. Хотя они не стандартизованы, тем не менее эти механизмы получили достаточно широкое распространение, а это означает, что, по крайней мере, один из них будет доступен в большинстве основных броузеров.

Практическое задание

Встроить cookie в страничку (например сохранять значение формы или какого-либо пользовательского выбора). Всю логику работы с cookie вынести в функции setCookie, getCookie, delCookie, свойство не должно обрабатываться непосредственно из обработчиков событий. Сделать механизм устойчивым ко вводу некорректных символов.