**Лабораторная работа 6 Работа с cookie**

*В работе 2 задания. Работа рассчитана на 2 пары.*

Для хранения небольших объёмов информации на диске компьютеров пользователя в браузере предусмотрен механизм cookie. Обычно применяется для хранения имени пользователя и пароля, а также информация о предыдущем посещении.

Механизм работы с cookies:



Клиент (веб-браузер) посылает серверу запрос (запрашивает у него страницу). Если в браузере есть cookies, связанные с этим сайтом, то он их посылает серверу в составе этого запроса.

Сервер получает запрос от клиента. Если в составе запроса есть куки, то их можно использовать для выполнения некоторой логики на сервере, подготовки пользователю персонализированной страницы или для чего-то другого. После этого отправляем клиенту ответ. В заголовке ответа отправляем веб-браузеру (клиенту) cookies, которые ему нужно будет сохранить.

Веб-браузер (клиент) получает ответ от сервера (страницу) и выводит его пользователю. Куки, которые пришли с сервера, браузер сохраняет в своё хранилище.

Записи cookie сохраняются в отдельных текстовых файлах, расположенных в папке cookies. Имя файла для записи cookie обычно образуется на основе имени пользователя и домена того сервера, ан котором создавался cookie-файл. Записи можно представить в виде строки, содержащей следующие:

имя записи;

содержание записи;

срок хранения (годности) записи;

домен сервера, который создал запись;

сведения о необходимости установки безопасного HTTP-соединения для доступа к записям;

расположение документов, которым разрешен доступ к записям.

Прочитать и установить cookies можно не только на стороне сервера, но и на клиенте - с помощью JavaScript. Осуществляется это посредством свойства cookie объекта document.

// получить значения всех cookies страницы

var cookies = document.cookie;

// выведем куки в консоль браузера

console.log(cookies);

// выведем с помощью функции alert

alert(cookies);

Формат строки

Запись данных в cookie файл с помощью javascript осуществляется выражением присваивания строки, содержащей cookie данные (document.cookie). При этом важно соблюдать формат строки:

document.cookie=”cookieName=данные

[; expires=строка\_времени\_GMT]

[; path=путь]

[; domain=домен]

[; secure]”

Квадратные скобки означают, что эти данные не обязательны. Если нужно сохранить пользователя STUDENT, то будет такой синтаксис:

document.cookie=”User\_Name=STUDENT”

Элемент cookie записи не должен содержать «точек с запятой», «запятых» и «пробелов». Избежать этого позволяет функция escape(), с помощью которой нужно предварительно обработать строку.

[; expires=строка\_времени\_GMT] — дата и время хранения cookie записи должны быть представлены строкой и содержать данные по Гринвичу

var expdate=new Date() // создаем объект даты

var monthFromNow=expdate.getTime() + (30\*24\*60\*60\*1000)

expdate.setTime(monthFromNow) //устанавливаем значение даты

document.cookie="User\_Name=STUDENT;

expires=" + expdate.toGMTString()

expdate=Thu, 30-Mar-70 00:00:01 GMT

[; path=путь] — означает в какой папке на компьютере будет хранится cookie запись

[; domain=домен] — выясняет домен текущего документа и помещает в cookie файл записи, соответствующие этому домену

[; secure] — принимает только true/false

Данные cookie, которые можно получить с помощью сценария на javascript, представляют собой единственную строчку — это значение свойства document.cookie.

Функция readCookie(name)

Функция readCookie(name) возвращает значение cookie-записи с именем name или null, если такая запись не найдена. В теле этой функции содержится вызов еще одной вспомогательной функции getCookieVal(j), возвращающей декодированное значение cookie-данных. Декодирование — встроенная функция unescape()

function readCookie(name) { // чтение cookie-данных записи

var xname=name + "=";

var xlen=xname.length;

var clen=document.cookie.length;

var i=0;

while(i < clen){

var j=i + xlen;

if (document.cookie.substring(i, j) == xname)

return getCookieVal(j);

i=document.cookie.indexOf(" ",1) + 1;

if (i == 0)

break;

}

return null;

}

Функция getCoolieVal()

function getCookieVal(n)

{

/\* вспомогательная функция, вызываемая из readCookie() \*/

var endstr=document.cookie.indexOf(";", n);

if (endstr == -1)

endstr=document.cookie.length;

return unescape(document.cookie.substring(n, endstr))

}

Функция writeCookie()

Она позволяет создать или обновить cookie-запись. Принимает следующие параметры: name, value, expires, path, domain, secure.

Параметры функции writeCookie()

name — строка, содержащая имя cookie-записи (обязательный параметр);

value — строка, содержащая значение cookie (обязательный параметр);

expires — объект даты (Date), содержащий срок хранения cookie-записи; если отсутствует, то после завершения работы браузера cookie-запись удаляется;

path — строка, содержащая путь cookie-записи; если параметр не указан, то используется путь вызванного документа;

domain — строка, содержащая домен нужной cookie-записи; если параметр не указан, то используется домен вызванного документа;

secure — логическое значение (true или false), определяющее необходимость использования безопасного HTTP-соединения.

function writeCookie(name, value, expires, path, domain, secure)

{

/\* запись cookie \*/

document.cookie =

name + "=" + escape(value) +

((expires) ? "; expires=" + expires.toGMTString(): "") +

((path) ? "; path=" + path : "") +

((domain) ? "; domain=" + domain : "") +

((secure) ? "; secure" : "")

}

Удаление cookie-записи

Созданные или измененные Cookie-записи будут записаны на диск только после закрытия браузера. До этого они существуют лишь в кеше, то есть в оперативной памяти.

function deleteCookie(name, path, domain)

{

/\* удаление cookie-записи \*/

if (readCookie(name))

{

document.cookie =

name + "=" + ((path) ? ";

path=" + path : "") + ((domain) ? ";

domain="+domain : "") + ";

expires=Thu, 30-Jan-70 00:00:01 GMT"

}

}

HTML-код для эксперимента со сценарием, записывающим и читающим cookie-запись:

<HTML>

<SCRIPT>

function readCookie(name) { // чтение cookie-данных записи

... // код

}

function getCookieVal(n){

// вспомогательная функция, вызываемая из readCookie()

... // код

}

function writeCookie(name, value, expires, path, domain, secure) {

// запись cookie

... // код

}

/ \* Срок хранения — 1 год от текущей даты: \*/

var d1=new Date();

var d2=d1.getTime() + (365\*24\*60\*60\*1000);

d1.setTime(d2);

/\* Запись и чтение cookie: \*/

writeCookie("myrecord","Привет!",d1);

alert("Содержимое записи cookie:\n"+readCookie("myrecord"))

</SCRIPT>

</HTML>

**Библиотека js-cookie**

js-cookie - библиотека JavaScript, предоставляющая методы для удобной работы с cookies.

Подключение js-cookie к странице

<script src="/path/to/js.cookie.js"></script>

Установка (set) cookie

Запись cookie осуществляется с помощью функции set:

// name - ключ (имя) куки

// value - значение, связанное с ключом name

// attributes (необязательный параметр) - атрибуты куки в формате объекта

Cookies.set('name', 'value'[,attributes]); Установить cookie для всех страниц сайта:

Cookies.set('nameCookie', 'valueCookie'); // => "nameCookie=valueCookie; path=/" Создать cookie, действительную 30 дней (от текущего момента времени) и видимую любыми страницами сайта:

Cookies.set('nameCookie', 'valueCookie', { expires: 30 }); // => "nameCookie=valueCookie; path=/; expires=Thu, 13 Apr 2017 13:00:15 GMT"

Выполнить запись куки, доступ к которой будет иметь только текущая страница (срок хранения 365 дней):

Cookies.set('nameCookie', 'valueCookie', { expires: 365, path: '' }); // => "nameCookie=valueCookie; expires=Wed, 14 Mar 2018 13:00:36 GMT"

Получение (get) куки

Чтение значения cookie осуществляется с помощью метода get:

Cookies.get('nameCookie'); // => "valueCookie"

Cookies.get('otherCookie'); // => undefined Получить все доступные cookies:

Cookies.get(); // => {nameCookie: "valueCookie"}

Удаление (remove) cookie

Для удаления куки предназначена функция remove:

Cookies.remove('nameCookie'); Удаление cookie с указанием атрибута path:

Cookies.set('name', 'value', { path: '' });

Cookies.remove('name', { path: '/' }); // не получится (не правильный путь)

Cookies.remove('name', { path: '' }); // удалится (указан правильный путь)

Важно! Когда вы удаляете cookie, вы должны указать тот же самый путь и домен, который использовался для установки cookie. Если атрибуты куки в Cookies.remove не указать, то будут браться те, которые заданы по умолчанию в Cookies.defaults (приведён ниже).

Например, удалим все cookies, имеющие в качестве пути значение '/':

path = '/';

var cookies = Cookies.get();

for (var cookie in cookies) {

Cookies.remove(cookie, { path: path });

}

**Установка атрибутов cookies**

Передача атрибутов при установке куки осуществляется посредством указания их в качестве значения последнего (3) аргумента функции Cookies.set.

Cookies.set('name', 'value', {

expires: 365,

path: '/news/',

domain: 'm.mydomain.ru',

secure: true

}); // => "name=value; path=/news/; expires=Wed, 14 Mar 2018 12:54:28 GMT; domain=m.mydomain.ru; secure"

expires - указывает длительность хранения cookie в браузере. Значение можно задавать как в формате числа (количество дней от текущего момента времени), так и ввиде даты. Если данный параметр не указать, то cookie будет ссесионным, т.е. он удалится после закрытия браузера.

Например, установим cookie на 30 дней (в качестве формата expires будем использовать дату):

// функция, возвращающая дату, которая будет через указанное количество дней от текущей

function getDateExpires(days) {

var date = new Date;

date.setDate(date.getDate() + days);

return date;

}

// запись cookie nameCookie=valueCookie

Cookies.set('nameCookie', 'valueCookie', { expires: getDateExpires(30) }); // => "nameCookie=valueCookie; path=/; expires=Thu, 13 Apr 2017 12:34:13 GMT"

path - строка, указывающая путь (по умолчанию: /), посредством которого будет определяться видимость (доступность) cookie. Cookie для некоторой страницы доступна только тогда, когда её path входит в адрес. По умолчанию куки видимы для всех страниц сайта.

Cookie, которая будет иметь в качестве пути текущее местоположение документа:

Cookies.set('nameCookie', 'valueCookie', { path: '' }); // => "nameCookie=valueCookie"

domain - строка, указывающая домен (по умолчанию: текущий), в котором cookie должна быть видна. При этом куки также будут доступны во всех поддоменах этого домена.

secure - устанавливает, необходимо ли при передаче cookies использовать безопасный протокол (https). По умолчанию: false (нет не требуется).

Кроме индивидуальной установки атрибутов куки при каждом вызове функции Cookies.set, их также можно назначить глобально посредством свойства defaults объекта Cookie.

Cookies.defaults = {

path: '/',

expires: 365

};

Cookies.set('nameCookie', 'valueCookie'); // => "nameCookie=valueCookie; path=/; expires=Wed, 14 Mar 2018 12:53:23 GMT"

Данные атрибуты будут использоваться функцией Cookies.set по умолчанию. Другими словами только тогда, когда вы вызываете функцию Cookies.set без последнего (третьего) параметра.

**Куки и формат JSON**

Библиотека js-cookie позволяет очень просто установить в качестве значения куки не только текстовое значение, но и данные в формате массива или объекта (а точнее их строковое представление посредством JSON.stringify):

Cookies.set('name', { name1: 'value1', name2: 'value2' });

Выполнять чтение cookie (имеющей в качестве значения JSON) с помощью функции Cookies.get не очень удобно, т.к. на выходе вы получите обычную строку (а не массив или объект).

Cookies.get('name'); // => "{"name1":"value1","name2":"value2"}"

Cookies.get(); // => {name:"{"name1":"value1","name2":"value2"}"}

Для этого можно воспользоваться функцией Cookies.getJSON. Данная функция не только получает cookie, но и разбирает её с помощью JSON.parse:

Cookies.getJSON('name'); // => {name1: "value1", name2: "value2"}

Cookies.getJSON('name')['name1']; // => "value1"

Cookies.getJSON('name')['name2']; // => "value2"

Cookies.getJSON(); // => {name: {name1:"value1", name2:"value2"}}

На сегодня cookie является сильным средством для сбора статистики в рекламных целях. Также с помощью куки можно отслеживать перемещения пользователя и его интересы.

**ЗАДАНИЕ 1**

Разработать страницу, при загрузке которой запрашивается логин и пароль. Сохранить это в куки.

Сделать запрос: показать секретные данные?

Вывести на страницу данные из файла куки текущего сеанса

Удалить ваше имя из куки

Закрыть браузер.

Можно воспользоваться текстом скрипта, приведенного на странице <https://myrusakov.ru/javascript-cookie.html>

<script type="text/javascript">

function setCookie(name, value) {

document.cookie = name + "=" + value;

}

function getCookie(name) {

var r = document.cookie.match("(^|;) ?" + name + "=([^;]\*)(;|$)");

if (r) return r[2];

else return "";

}

function deleteCookie(name) {

var date = new Date(); // Берём текущую дату

date.setTime(date.getTime() - 1); // Возвращаемся в "прошлое"

document.cookie = name += "=; expires=" + date.toGMTString(); // Устанавливаем cookie пустое значение и срок действия до прошедшего уже времени

}

setCookie("firstname", "Вася"); // Устанавливаем cookie

alert(getCookie("firstname")); // Выводим cookie

deleteCookie("firstname"); // Удаляем cookie

alert(getCookie("firstname")); // Убеждаемся, что ничего не осталось

</script>

**ЗАДАНИЕ 2**

1. Разработать страницу , на которой разместить ваше фото, имя и перечень ваших хобби, например, хоккей зимой, футбол летом, или вязание зимой, плавание летом.

При посещении страницы, в зависимости от сезона показывать картинку с тем видом хобби, которое соответствует сезону. Добавить приветствие, запросив имя при открытии страницы. Сопровождая примерно таким текстом при последующих открытиях:

О, привет, Наталья! Сегодня плаваем, так как наступило лето.