Работа с Web хранилищами, на примере корзины для покупок

 13.12.2013

 Дмитрий Кочетов

 HTML, jQuery

Google+

3

Twitter

Вконтакте

1

Facebook

Здравствуйте, уважаемые читатели XoZbloga! Не так давно я уже писал пост о локальном хранении пользовательских данных, с помощью технологии Web SQL Database. В этом уроке продолжу данную тему и разберу на примере, корзины для покупок, способ работы с web хранилищами.

Демо 

 Скачать

Что такое web хранилище?

Технология Web хранилищ позволяет хранить данные в браузере пользователя, похожая на cookie (куки), однако с расширенными возможностями. Изначально была частью спецификации HTML5, но сейчас является самостоятельной спецификацией. На данный момент поддерживается всеми современными браузерами:











Существует два типа web хранилищ:

Local Storage (Локальное хранилище) — хранит данные постоянно. И это тот тип, который мы будем использовать.

Session Storage (Сессионное хранилище) — хранит данные в течение одной сессии. Пока открыта вкладка браузера, данные хранятся.

Структура хранилища состоит из пар ключ (key) — значение (value).

HTML разметка

Заострять особое внимание на разметке не будем. Работа с локальным web хранилищем рассматривается на примере импровизированной корзины покупок посетителя интернет-магазина. Поэтому нам понадобится картинка товара, поле для ввода количества и кнопка для добавления в корзину. Следующий код представляет собой фрагмент блока с одним товаром:

<!-- Блок с товаром -->

<div class="tov">

<!-- Изображение товара -->

<img src="img/1.jpg"><br/>

<!-- Поле для ввода количества -->

<input type="text" id="pr1">&nbsp;

<!-- Кнопка добавления в корзину (в локальное хранилище) -->

<button data-pr="pr1" class="incart">В корзину</button>

</div>

Стоит обратить внимание на атрибут id у поля для ввода и атрибут data-pr у кнопки, они идентичны. То есть для каждого поля ввода своя кнопка. Ну и конечно подразумевается, что формирование разметки происходит автоматически (в цикле) на основании данных из базы. Поэтому трудностей с этим, возникнуть не должно.

Под блоком с товарами, расположим корзину, пусть это будет маркированный список:

<ul class="cart"></ul>

Использование интерфейса локального хранилища

Спецификация Web Storage описывает два объекта хранения, local storage и session storage, оба доступны через глобальные переменные LocalStorage и sessionStorage, которые определены на уровне окна (window level).

Проверка совместимости

Перед тем как приступить к работе с хранилищем, необходимо проверить на совместимость с этой технологией браузер пользователя. Для этого можно использовать следующий JavaScript код:

// Функция которая возвращает истину если браузер поддерживает локальные хранилища

function web\_storage() {

try {

return 'localStorage' in window && window['localStorage'] !== null;

} catch (e) {

return false;

}

}

if(web\_storage()){

// есть поддержка localStorage

}else{

// нет поддержки localStorage

}

Следующие методы доступны как для локального так и для сессионного хранилища, однако рассматривать будем на примере первого.

Добавление/изменение данных в хранилище

Чтобы изменить значение или добавить новое значение, используется метод setItem():

localStorage.setItem("myData", "Мои данные");

Первый аргумент является уникальным ключом, который идентифицирует данные, а второй аргумент непосредственно данные, которые Вы хотите добавить. Кроме того, Вы можете использовать и индекс элемента массива для добавления данных:

localStorage["myData"] = "Мои данные";

// или...

localStorage.myData = "Мои данные";

В браузере Google Chrome, открыть панель ресурсов можно нажав F12 и перейдя на закладку Resources.

Получение данных из хранилища

После того как в хранилище появились данные, их можно запросить с помощью метода getItem():

var data = localStorage.getItem("myData");

В качестве аргумента указан ключ значения. Даже если вы закрываете браузер, перезагрузите страницу, или используете этот метод с другой страницы (на том же домене), все равно получите занесенные данные. Альтернативный вариант получения данных:

var data = localStorage["myData"];

// или ...

var data = localStorage.myData;

Удаление данных

Если Вам нужно удалить сохраненное значение из объекта хранения, вы можете сделать это с помощью метода removeItem():

localStorage.removeItem("myData");

Либо полностью очистить локальное хранилище, с помощью метода clear():

localStorage.clear();

Хранение сложных типов данных

Web Storage ограничивается хранением только строковых значений, так что не возможно хранить другие типы данных без их преобразования в строку. Однако можно легко обойти это ограничение, если использовать кодирование данных в виде JSON:

var data = {

key1 : "string", // строка

key2 : true, // булево

key3 : [1,2.3] // массив

};

localStorage.setItem("myData", JSON.stringify(data));

При чтении нужно не забыть декодировать данные обратно:

var data = JSON.parse(localStorage.getItem("myData"));

Перебор данных хранилища

Для того, чтобы определить какое количество пар ключ — значение вы храните в local storage, используется свойство length:

var numValues = localStorage.length;

Метод key() принимает один аргумент — индекс от 0 до length-1, и возвращает имя ключа в этом положении:

var data = localStorage.key(0); // имя ключа с индексом 0

Однако нет никакой гарантии, что ключи находятся в порядке их добавления, но метод все еще может быть полезен, если Вам нужно перебрать все сохраненные значения:

// перебираем все значения в цикле

for (var i=0,key,value; i < localStorage.length; i++) {

// получаем имя ключа

key = localStorage.key(i);

// получаем значение по имени ключа

value = localStorage.getItem(key);

// выводим в консоль

console.log(key, value);

}

jQuery

Последнее, что осталось сделать, так это применить полученные знания на практике Разметка у нас есть, осталось описать обработчики событий при добавлении данных в локальное хранилище и вывод их на страницу в «корзину покупателя». Давайте начнем с описания функции вывода данных из хранилища на страницу:

// Отобразим содержимое хранилища

function ref\_cart() {

var output = "";

// Сначала очистим маркированный список

$(".cart li").remove();

// Перебором получаем все данные из хранилища

for (var i = 0; i < localStorage.length; i++) {

output += "<li>ID: "+localStorage.key(i)+" | Количество: "+localStorage.getItem(localStorage.key(i))+" <button data-pr='"+localStorage.key(i)+"' class='remove'> X </button></li>";

}

// Добавляем на страницу

$(".cart").append(output);

}

Хорошо, функцию для получение данных и отображение их на странице создали. Теперь можно перейти и к созданию обработчика добавления данных в хранилище. Для этого у нас используется кнопки, поэтому отслеживаем клик:

// Добавить в корзину

$(".incart").on('click', function() {

var reg = /[0-9]/, // регулярное выражение для проверки вводимых символов

id = $(this).attr("data-pr"), // идентификатор продукта

kolvo = $("#"+id).val(); // введенное количество товара

// проверяем, количество должно быть только числом

if (reg.test(kolvo)) {

// проверяем совместимость

if(web\_storage()) {

// очищаем поле с количеством

$("#"+id).val('');

// добавляем данные в локальное хранилище

localStorage.setItem(id, kolvo);

// вызываем функцию отображения корзины на странице

ref\_cart();

} else{

alert("Ваш браузер не может работать с локальным хранилищем!");

}

} else {

$("#"+id).val('');

alert("Использовать только числа!");

}

});

Последним штрихом, можно считать обработчик удаления одного из элементов в корзине. Для этого у нас есть кнопки, у каждого элемента списка, вот код обработки:

// Удалить из корзины

$(document.body).on('click','.remove',function() {

// Удаляем из хранилища

localStorage.removeItem($(this).attr("data-pr"));

// Удаляем элемент списка

$(this).parent('li').remove();

});

На этом все, не забудьте скачать исходники (кнопка в начале страницы). Спасибо за прочтение, надеюсь информация изложенная в этом уроке Вам пригодится.

Источник статьи/урока: http://xozblog.ru

По материалам: http://www.codeproject.com/Articles/361428/HTML5-Web-Storage

Правила перепечатки

Чтобы оставаться в курсе свежих статей и уроков подписывайтесь на еженедельную почтовую рассылку или на новостную ленту RSS. Спасибо!

Похожие статьи: