**Лабораторная работа 10 Технология AJAX**

**Цель работы:**

***изучить основы технологии AJAX.***

Написать клиентское и серверное приложение обменивающиеся асинхронными данными.

**Краткие теоретические сведения**

AJAX —набор техник разработки веб-интерфейсов,позволяющих делать динамические запросы к серверу без видимой перезагрузки веб-страницы: пользователь не замечает, когда его браузер запрашивает данные.

AJAX обеспечивает динамичность и асинхронность web-разработок при отсутствии необходимости обновления страниц.

AJAX. обеспечивает динамическое и асинхронное поведение, где исключается обновление страниц. Посредством AJAX пользователь может взаимодействовать с web-

страницами, подобно работе клиентов с более богатыми возможностями.

Основа - объект XMLHttpRequest,

Самый яркий пример - Google Maps.

Пользуясь им, можно находить определенную местность на карте планеты, затемпереходить к более мелким объектам, прокручивать перетягивать карту без необходимости обновления страницы.

AJAX — аббревиатура от Asynchronous JavaScript and XML (и DHTML, и

т.д.). (как это представил Джис Джеймс Гаррет (Jesse James Garrett), он первым ввел

термин 'AJAX' для асинхронного JavaScript + XML). Т.е. это коллекция технологий,

существующих с момента появления Web.

Возможности, предоставляемые AJAX:

Стандартно-базированная презентация с использованием XHTML и CSS;

Динамическое отображение и взаимодействие с использованием

объектной модели документа;

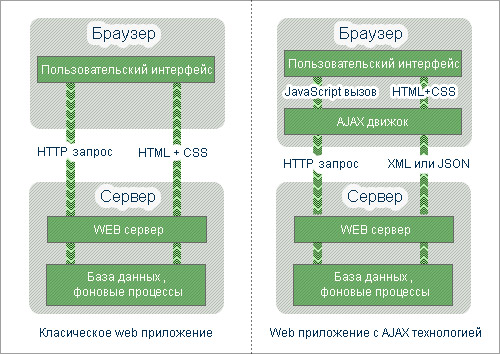
Взаимообмен данными и манипуляция с задействованием XML и XSLT;

Асинхронное извлечение данных с использованием XMLHttpRequest;

JavaScript, связывающий все вместе.

AJAX позволяет писать быстрореагирующие веб-приложения, в которых не нужно постоянно обновлять страницы. AJAX поддерживается всеми браузерами.

Как работает AJAX



AJAX — это технология, которая обрабатывает операции в JavaScript и асинхронно запускает на стороне сервера операции, предоставляющие желаемый результат.

В основе технологии AJAX лежит объект XMLHttpRequest. Изначально он

появился в Internet Explorer, а затем — в Mozilla/Safari.

Объект XMLHttpRequest

Создание объекта

.

<script language="javascript" type="text/javascript">

var request = new XMLHttpRequest();

</script>

Это объект, который управляет всем взаимодействием с сервером –

, который общается с сервером.

В нормальных Web-приложениях пользователи заполняют поля форм и

нажимают кнопку Submit (подтвердить). Затем форма передается на сервер полностью, сервер обрабатывает сценарий (обычно PHP или Java, возможно, CGI- процесс или что-то в этом роде), а потом передает назад всю новую страницу. Эта страница может быть HTML-страницей с новой формой с некоторыми заполненными данными, либо страницей подтверждения, либо, возможно, страницей с какими-то выбранными вариантами, зависящими от введенных в оригинальную форму данных.

Пока сценарий или программа на сервере не обработается и не возвратится новая форма, пользователи должны ждать. Их экраны очистятся и будут перерисовываться по мере поступления новых данных от сервера. Вот где проявляется низкая интерактивность – пользователи не получают немедленной обратной реакции и определенно чувствуют себя не так, как при работе с настольными приложениями.

Ajax по существу помещает технологию JavaScript и объект XMLHttpRequest

Между Web-формой и сервером. Когда пользователи заполняют формы, данные передаются в какой-то JavaScript-код, а не прямо на сервер. Вместо этого JavaScript- код собирает данные формы и передает запрос на сервер. Пока это происходит, форма на экране пользователя не мелькает, не мигает, не исчезает и не блокируется. Другими словами, код JavaScript передает запрос в фоновом режиме; пользователь даже не замечает, что происходит запрос на сервер. Более того, запрос передается асинхронно, а это означает, что JavaScript-код (и пользователь) не ожидают ответа сервера. То есть, пользователи могут продолжать вводить данные, прокручивать страницу и работать с приложением. вашей

Затем сервер передает данные обратно в JavaScript-код (все еще находящийся в Web-форме), который решает, что делать с данными. Он может обновить поля формы "на лету", придавая свойство немедленности вашему приложению – пользователи получают новые данные без подтверждения или обновления их форм. JavaScript-код может даже получить данные, выполнить какие- либо вычисления и передать еще один запрос, и все это без вмешательства пользователя!

Добавление JavaScript-кода

JavaScript-код используется для управления DOM и для работы со структурой HTML-формы и всеми XML-данными, возвращаемыми сервером.

. Сбор и установка значений полей при помощи JS-кода

// Получить значение поля "phone" и записать его в переменную phone

var phone = document.getElementById("phone").value;

// Установить значения в форме, используя массив response

document.getElementById("order").value = response[0];

document.getElementById("address").value = response[1];

Далее HTML+DOM

Получение объекта Request

XMLHttpRequest является центральным для Ajax-приложений

Увы надо помнить об особенностях разных браузеров.

Работа с браузерами Microsoft

Браузер Microsoft Internet Explorer для обработки XML использует анализатор

MSXML.

Версия MSXML зависит от версии технологии JavaScript, установленной в Internet Explorer, поэтому нужно написать код, подходящий для обеих версий.

Создание объекта XMLHttpRequest в браузерах Microsoft

var request = null;

function createRequest () {

try {

request = new ActiveXOblect("Msxml2.request");

} catch (othermicrosoft) {

try {

request = new ActiveXObject("Microsoft.request");

} catch (failed) {

request = null;

}

}

}

request = new ActiveXObject("Msxml2.request");

и

request = new ActiveXObject("Microsoft.request");

.

Запрос на сервер . test.php ( можно попробовать в NODE)

<?php

//Sending zipCode

echo rand(10000, 99999);

?>

Обработка ответа

ответ сервера.

Не делать ничего, пока свойство

request.readyState

не будет равно 4.

•

Сервер будет записывать свой ответ в свойстве

request.responseText

.

Первый момент (состояния готовности) – о нем вы должны сами разобраться,

т.е. уточнить какие бывают стадии HTTP-запроса. Пока вы просто проверяйте на

равенство определенному значению (4), и все будет работать. Второй момент

(использование свойства request.responseText для получения ответа от сервера)

является простым.

Обработка ответа от сервера

function updatePage() {

if(request.readyState == 4) {

var response = request.responseText;

//Что-то делаем с пришедшими нам данными, например, присваиваем

их полю с id="zipCode"

document.getElementById("zipCode").value = response;

}

}

Опять же, код не является трудным или сложным. Он ожидает, пока сервер не

вызовет его с нужным состоянием готовности, и затем использует значение, которое

сервер возвращает, для установки другого поля формы. В результате поле zipCode

неожиданно появляется с ZIP-кодом, но пользователь ни разу не щелкнул по кнопке

!

Это поведение настольного приложения, о чем мы говорили ранее.

Вы, возможно, заметили, что поле zipCode является обычным текстовым полем.

После возврата сервером ZIP-кода и установки этого поля методом updatePage()

В значение ZIP-кода города/штата пользователи могут переопределить это значение.

Так сделано умышленно по двум причинам: сохранить этот пример простым и

показать вам, что иногда нужно, чтобы пользователи имели возможность

переопределить значения, возвращенные сервером. Помните об обоих моментах; они

важны при хорошем дизайне пользовательского интерфейса.

**Перехват в Web-формахЕсть**

**Есть JS метод** собирающий введенную пользователем в форму информацию, передает ее серверу,предоставляете еще один JavaScript-метод для обработки ответа устанавливает значение поля, когда этот ответ приходит. Все что осталось на самом деле – вызвать

этот первый метод и запустить полный процесс.

. Запуск Ajax-процесса

<form>

<p>City: <input type="text" name="city" id="city" size="25"

onChange="getResult();" /></p>

<p>Zip Code: <input type="text" name="zipCode" id="zipCode" size="5"

readonly /></p>

</form>

Для более легкой работы с HTML при помощи JavaScript были разработаны

несколько фреймворков, которые упрощают повседневные рутинные функции и

помогают программисту не задумываться о реализации примитивов, например,

доступа к полям DOM-а, их изменение стандартными средствами JavaScript, а

предоставляют удобные инструменты для данных функций.

jQuery

jQuery — библиотека JavaScript, фокусирующаяся на взаимодействии JavaScript

и HTML. Библиотека jQuery помогает легко получать доступ к любому элементу

DOM, обращаться к атрибутам и содержимому элементов DOM, манипулировать ими.

Также библиотека jQuery предоставляет удобный API по работе с Ajax. Т.к. jQuery

является более простой для изучения остановимся на ней подробнее.

Возможности:

▪

переход по дереву DOM, включая поддержку Xpath как плагина;

▪

события;

▪

визуальные эффекты;

▪

AJAX-дополнения;

▪

JavaScript-плагины

Философия

Точно так же, как CSS отделяет визуализацию от структуры HTML, JQuery

отделяет поведение от структуры HTML. Например, вместо прямого указания на

обработчик события нажатия кнопки, управление передаётся JQuery, идентифицирующей кнопки и затем преобразовывающий его в обработчик события клика. Такое разделение поведения и структуры также называется принципом ненавязчивого JavaScript.

Библиотека jQuery содержит функционал, полезный для максимально широкого круга задач.

Подключение фрейморка jQuery

<script type="text/javascript" src="jquery.js"></script>

Добавим в  **форму дополнительное поле, в котором будем изменять**

**значение парам**етра учета количества сделанных изменений:

Добавление дополнительного поля в форму

<form>

<p>City: <input type="text" name="city" id="city" size="25"

onChange="getResult();" /></p>

<p>ZipCode: <input type="text" name="zipCode" id="zipCode" size="5"

readonly /> получен с помощью AJAX запроса</p>

<p>Значение было изменено <input type="text" name="jq" id="jq"

size="3" value="0" readonly /> раз с помощью jQuery</p>

</form>

Дополним функцию updatePage() вызовом дополнительной функции jq(),

которая, используя механизмы jQuery, будет изменять данные в поле с id=”jq”, в

котором указывается количество обновлений страницы:

Исходный код функции jq()

function jq() {

//Записываем в переменную X значение поля с id='jq'

$x = $("#jq").attr("value");

//Увеличиваем переменную на 1

$x++;

//Обновляем значение атрибута value поля с id='jq'

$("#jq").attr("value", $x);

}

**Порядок выполнения работы**

Ознакомится с теоретическими сведениями

2. Изучить работу скриптов, приведенных в теоретических сведениях.

3.Релизовать задание согласно варианта.

4. Составить отчет о выполнении работы.

Варианты заданий **Внимание!!! Страница не перегружается при переходах**

1.Проверка логина при регистрации – проверка на существование вводимого логина в базе данных пользователей, при наборе логина в соответствующем поле.

2.Корзина товаров в онлайн магазине. Добавление/удаление.

3.Города и области – выборка области по выбранному городу.

4.Подсчет хеша при заполнении строки.

5.Транслитерация.

6.Простейший почтовый клиент – выборка темы письма по запросу и по клику на теме письма доставка его содержимого.

7.Мониторинг сервисов на ПК. Опрос списка запущенных процессов, и асинхронное обновление.

8.Гостевая книга, Комментарии – добавление/удаление.

9.Чат.

10. <https://sites.google.com/site/webkss2kurs/zacetnaa-rabota/sozdanie-prostogo-ajax-sajta-s-pomosu-jquery> - изменить сайт по своему усмотрению, добавить авторский дизайн и перегрузку страниц

11 Студент/номер зачетной книжки – в зачетной книжке зашифрован курс и специальность. Выбираем по курсу и специальности .

Оригинал статьи

http://www.ibm.com/developerworks/web/library/wa-

ajaxintro1.html?S\_TACT=105AGX99&S\_CMP=CP

2.

AJAX из викиучебника –

http://ru.wikibooks.org/wiki/AJAX

Официальный сайт jQuery – jquery.com

4.Страница в wikipedia -

<http://ru.wikipedia.org/wiki/JQuery>

<https://studfiles.net/preview/1439239/>

<https://www.youtube.com/watch?v=AnjP0tuq700>

<https://www.ibm.com/developerworks/ru/library/wa-ajaxintro1/>

<https://metanit.com/web/jquery/6.1.php>

<https://wiki.rookee.ru/ajax/>

<http://bourabai.kz/php/ajax.htm>

<https://learn.javascript.ru/ajax-intro>

<http://javascript.ru/ajax/intro>

<https://habrahabr.ru/post/14246/>