**Создание AJAX запросов в jQuery**

С помощью JavaScript Вы можете создавать асинхронные запросы и отправлять их на сервер.

Использование асинхронных запросов позволяет значительно ускорить загрузку страниц, так как в этом случае обновляться будет только та часть страницы, которая содержит новые данные, а не страница целиком.

Техника использования асинхронных запросов называется **AJAX** - **A**synchronous **J**avaScript **A**nd **X**ML (Асинхронный JavaScript и XML).

Создание AJAX запросов на "чистом" JavaScript имеет несколько недостатков:

* Код даже самого простого AJAX запроса получается достаточно громоздким и сложным для понимания без специального ознакомления.
* Необходимо добавлять дополнительный код для поддержки старых версий браузеров.

AJAX запрос созданный с помощью jQuery занимает всего одну строчку кода, и уже оптимизирован для использования и с новыми и старыми версиями браузеров.

**Синтаксис:**

$("селектор").load(url,данные,функция)

**селектор** выбирает элемент, в котором будет отображен результат запроса.

**url** адрес файла, который будет запрошен у сервера с помощью AJAX.

**данные** которые будут переданы запрошенному файлу. Если Вы хотите запросить файл, при этом не передавать никаких данных, оставьте данный параметр пустым.

**функция** имя функции, которая будет вызвана после выполнения запроса.

**Пример:**

$(document).ready(function(){

$("#but1").click(function(){

$("#par1").load("testfile.txt");

})

});

Теперь приведем пример запроса, который пересылает данные.

В примере ниже мы пересылаем скрипту add.php переменную x со значением 4 и переменную y со значением 5 скрипт в свою очередь принимает эти переменные, производит их сложение и отправляет результат обратно.

**Пример:**

$(document).ready(function(){

$("#but1").click(function(){

$("#par1").load("add.php","x=4&y=5");

})

});

# Сопровождающие функции

Сопровождающие функции позволяют выполнять различный код в зависимости от состояния выполнения AJAX запроса.

Метод **ajaxSend()** выполняет переданный в него код во время отправления AJAX запроса.

Метод **ajaxComplete** выполняет переданный в него код после завершения выполнения AJAX запроса (*при этом неважно произошла ошибка или нет*).

**Пример:**

$(document).ready(function(){

$("#par1").ajaxSend(function(){

$("#img1").css("display","block");

})

$("#par1").ajaxComplete(function(){

$("#img1").css("display","none");

})

$("#but1").click(function(){

$("#par1").load("add.php","x=10&y=100");

})

});

Метод **ajaxSuccess** выполняет переданный в него код, если выполнение AJAX запроса завершается успешно.

Метод **ajaxError** выполняет переданный в него код, если выполнение AJAX запроса завершается с ошибкой.

**Пример:**

$(document).ready(function(){

$("#par1").ajaxSend(function(){

$("#img1").css("display","block");

});

$("#par1").ajaxComplete(function(){

$("#img1").css("display","none");

});

$("#par1").ajaxError(function(){

alert("Выполнение AJAX запроса завершено с ошибкой.\nПроверьте Ваш запрос

на наличие ошибок и попробуйте отправить его еще раз.");

});

$("#par1").ajaxSuccess(function(){

alert("AJAX запрос успешно выполнен!");

});

$("#but1").click(function(){

$("#par1").load("add.php","x=10&y=100");

});

$("#but2").click(function(){

$("#par1").load("abb.php","x=10&y=100");

});

});

Методы **ajaxStart** и **ajaxStop** используются для сопровождения группы AJAX запросов.

Метод **ajaxStart** выполняет переданный в него код при отправлении первого запроса из группы.

Метод **ajaxStop** выполняет переданный в него код при завершении последнего запроса из группы.

**Пример:**

$(document).ready(function(){

$("#par1").ajaxStart(function(){

$("#wrap").css("display","block");

});

$("#par1").ajaxStop(function(){

$("#wrap").css("display","none");

});

$("#but1").click(function(){

$("#par1").load("add.php","x=5&=10");

$("#par1").load("add.php","x=10&y=100");

$("#par1").load("testfile.txt");

$("#par1").load("add.php","x=45&y=32");

});

});

# Низкоуровневые AJAX запросы

Высокоуровневые методы создания AJAX запросов (т.е. методы которые были разобраны ранее в данной главе) подходят для создания большей части запросов, но иногда их функциональности становится недостаточно.

Низкоуровневые AJAX запросы предоставляют более широкую функциональность, но более сложны в использовании.

**Синтаксис высокоуровневого AJAX запроса:**

$("#par1").load("add.php","x=4&y=5")

**Синтаксис аналогичного низкоуровневого AJAX запроса:**

$("#but1").click(function(){

$.ajax({url:"add.php",

data:"x=4&y=5",

success:function(result){

$("#par1").html(result)}

});

});

**Сделайте сами**

**Задание** реализуйте подпункты перечисленные ниже путем добавления на страницу соответствующего jQuery кода:

1. Создайте AJAX запросы, которые должны после нажатия на соответствующие кнопки обращаться к файлам autumn.txt, winter.txt, summer.txt, и spring.txt и затем отображать результат запроса в абзац с id=par1.
2. После нажатия на кнопку с id=but5 элемент span c id=let1 должен содержать первую букву кодового слова, span c id=let2 вторую и т.д. (подробнее задание данного пункта объяснено в редакторе).
3. Выполните предыдущий пункт используя низкоуровневый метод .ajax(). После нажатия на кнопку с id=but6 элемент span c id=slet1 должен содержать первую букву кодового слова, span c id=slet2 вторую и т.д.