Владимир Сабанцев, Юрий Кучма

МЕТОДИЧЕСКИЕ УКАЗАНИЯ «jQuery»

Пишем простейшие элементы на jQuery



СОДЕРЖАНИЕ

[Подключение](#h.gjdgxs)

[Выборка элементов](#h.30j0zll)

[Методы манипуляции элементами](#h.1fob9te)

[Методы анимации объектов](#h.3znysh7)

[Обработчики событий jQuery](#h.2et92p0)

[Пример: Таймер](#h.tyjcwt)

[Пример работы с изображениями](#h.3dy6vkm)

[Полезные ссылки](#h.1t3h5sf)

**Подключение**

**jQuery** - это библиотека, которая значительно упрощает и ускоряет написание JavaScript кода.

Девиз jQuery *"write less, do more"* (пиши меньше, делай больше) отражает ее главное предназначение.

jQuery позволяет создавать анимацию, обработчики событий, значительно облегчает выбор элементов в DOM и создание AJAX запросов.

Данная библиотека работает со всеми браузерами (*IE 6.0+, FF 2.0+, Safari 3.0+, Opera 9.0+, Chrome*). Это значит, что Вам больше не нужно будет беспокоиться о кроссбраузерной совместимости JavaScript кода.

Для jQuery написано огромное количество плагинов, которые позволяют расширить ее возможности еще больше.

Для того, чтобы начать использовать jQuery необходимо:

1. Скачать ее с [официального сайта](http://jquery.com/). Существуют две версии jQuery: для использования в готовых приложениях (production) и для разработки (development). Версия для разработки содержит комментарии и структурированный код. В сокращенной версии нет комментариев и код в ней не структурирован зато она занимает меньше места и поэтому страницы с ней будут загружаться быстрее. После того, как Вы выберите подходящую версию нажмите на кнопку "Download ( jQuery )"
2. Добавить ее на страницу. Для этого следующий код должен быть добавлен на страницу в секцию head:

Подключение

<html>

<head>

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

</head>

<body>

<input type="button" value="Нажмите на меня" />

<script>

$(document).ready(function(){

$("input").click(function(event){

alert("Поздравляем! Вы починили код!");

});

});

</script>

</body>

</html>

Код jQuery как и код JavaScript состоит из последовательно идущих команд. Команды являются основной структурной единицей jQuery.

Стандартный синтаксис jQuery команд:

$(селектор).метод();

Знак $ сообщает, что символы идущие после него являются jQuery кодом;

Селектор позволяет выбрать элемент на странице;

Метод задает действие, которое необходимо совершить над выбранным элементом. Методы в jQuery разделяются на следующие группы:

* Методы для манипулирования DOM;
* Методы для оформления элементов;
* Методы для создания AJAX запросов;
* Методы для создания эффектов;
* Методы для привязки обработчиков событий.

**Выборка элементов**

Прежде, чем проводить с элементами страницы, какие-то действия, их нужно отобрать. В данной статье собраны всевозможные способы отбора элементов, которые применяются в практике написания скриптов с помощью фреймворка jQuery.

*Базовые селекторы*

$( " \* " ) //выбор абсолютно всех элементов присутствующих в текущем html-документе

$( " #mydiv " ) // выбор элемента на странице по названию уникального идентификатора (будет отобран элемент с id="mydiv")

$( " .myclass " ) //выбор элементов на странице по названию класса (будут отобраны все элементы, которым присвоен класс "myclass")

$( " p " ) // выбор элементов на странице по названию тега (будут отобраны все параграфы, находящиеся на странице)

*Иерархические селекторы*

$( " div.demo span " ) // выбор всех span, находящихся внутри блока с классом demo (будут отобраны все "span", которые находятся внутри "div" с классом "demo", если в span будет вложен еще один span, то он тоже будет отобран)

$( " div ", " #myid " ) // то же, что и в предыдущих примерах (будут отобраны все "div", которые находятся внутри блока с идентификатором "myid")

$( " div.myclass > span " ) // выбор дочерних элементов (будут отобраны все span, находящийся в блоке с классом "myclass", и являющиеся дочерними (потомками первого уровня) по отношению к этому блоку, если в span будет вложен еще один span, то он не будет отобран)

$( " #demo + p " ) // выбор следующего за указанным элемента (будет отобран параграф, находящийся сразу же за элементом с идентификатором "demo")

$( " а ~ p " ) // выбор всех, следующих за указанным элементов (будут отобраны все параграфы, находящийся за ссылкой и являющиеся сестринскими по отношению к этой ссылке)

*Селекторы форм*

$( " :input " ) // выбор всех элементов input

$( " input:text " ) // выбор всех элементов input с type="text"

$( " input:password " ) // выбор всех элементов input с type="password"

$( " input:radio " ) // выбор всех элементов input с type="radio"

$( " input:checkbox " ) // выбор всех элементов input с type="checkbox"

$( " input:reset " ) // выбор всех элементов input с type="reset"

$( " input:submit " ) // выбор всех элементов input с type="submit"

$( " input:button " ) // выбор всех элементов input, являющихся кнопками (кнопки с атрибутом button, или кнопки созданные с помощью тегов button)

$( " input:file " ) // выбор всех элементов input с type="file"

*Базовые фильтры*

$( " div:odd " ) //выбор нечетных блоков "div" на странице (будут отобраны только нечетные элементы "div", присутствующие на странице - отсчет начинается с нуля)

$( " #mytable tr:even " ) //выбор четных "tr" в таблице (будут отобраны только четные элементы "tr" из таблицы с идентификатором "mytable" - отсчет начинается с нуля)

$( " img:not(#mypic img) " ) //выбор всех элементов на странице кроме указанных (будут отобраны все элементы "img", присутствующие на странице, кроме изображений, расположенных в блоке с идентификатором "mypic")

$( " div:eq(3) " ) //выбор элемента по порядковому номеру (будут отобран четвертый по счету блок "div" на текущей странице - отсчет начинается с нуля)

$( " div:gt(1) " ) //выбор элементов c индексом большим, чем указано в параметре (будут отобраны все блоки "div" на текущей странице начиная с третьего - отсчет начинается с нуля)

$( " div:lt(3) " ) //выбор элементов c индексом меньшим, чем указано в параметре (будут отобраны все блоки "div" на текущей странице до третьего включительно - отсчет начинается с нуля)

$( " :header " ) //выбор всех заголовков на текущей странице

$( " div:animated " ) //выбор всех блоков "div", анимированных в текущий момент времени

$( " p:first " ) //выбор первого абзаца в текущем документе

$( " p:last " ) //выбор последнего абзаца в текущем документе

*Фильтры контента*

$( " p:contains ( текст ) " ) //выбор элементов в текущем документе в содержимом которых встретится текст, переданный параметром в фильтр (будут отобраны все параграфы содержащие внутри себя слово "текст")

$( " div:empty " ) //выбор всех пустых элементов (будут отобраны все блоки "div" на текущей странице, не содержащие внутри себя ничего, включая текст)

$( " div:parent " ) //выбор всех элементов содержащих внутри себя дочерние элементы (будут отобраны все блоки "div" на текущей странице, внутри которых есть дочерние элементы, включая текст)

$( " div:has( span.red ) " ) //выбор всех элементов, которые содержат внутри себя хотя бы один из элементов переданный в параметре фильтра (будут отобраны все блоки "div" на текущей странице, содержащие внутри себя хотя бы один span с классом red)

*Фильтры видимости*

$( " div:hidden " ) //выбор всех скрытых элементов на текущей странице

$( " div:visible " ) //выбор всех видимых элементов на текущей странице

*Фильтры атрибутов*

$( " [alt] " ) //выбор элементов на текущей странице, содержащих указанный атрибут (будут отобраны все элементы на текущей странице, у которых есть атрибут alt)

$( " [alt=myatr] " ) //выбор элементов на текущей странице, содержащих указанный атрибут с указанным значением (будут отобраны все элементы на текущей странице, у которых есть атрибут alt со значением myatr)

$( " [alt!=myatr] " ) //выбор элементов на текущей странице, содержащих указанный атрибут, но у которых значение не равно указанному (будут отобраны все элементы на текущей странице, у которых есть атрибут alt, содержащие значение не myatr)

$( " [alt^=myatr] " ) //выбор элементов на текущей странице, содержащих указанный атрибут, у которых значение начинается с указанного (будут отобраны все элементы на текущей странице, у которых есть атрибут alt и значение будет начинаться со слова myatr)

$( " img[src\*=pic] " ) // выбор элементов по части значения атрибута (будут отобраны все изображения, находящийся на странице, у которых в атрибуте "src" встречается слово "pic")

$( " [alt$=myatr] " ) //выбор элементов на текущей странице, содержащих указанный атрибут, у которых значение заканчивается указанным (будут отобраны все элементы на текущей странице, у которых есть атрибут alt и значение будет заканчиваться словом myatr)

*Фильтры семейств*

$( " div:first-child " ) // выбор первого дочернего элемента (будет отобран первый дочерний элемент в блоке div на текущей странице)

$( " div:last-child " ) // выбор последнего дочернего элемента (будет отобран последний дочерний элемент в блоке div на текущей странице)

$( " div:only-child " ) // выбор единственного дочернего элемента (будет отобран единственный дочерний элемент в блоке div на текущей странице, который содержит этот элемент - если в блоке будет содержаться несколько дочерних элементов, то ничего отобрано не будет)

$( " div:nth-child(even) " ) // выбор четных дочерних элементов (будут отобраны все четные дочерние элементы в блоке div на текущей странице)

$( " div:nth-child(odd) " ) // выбор нечетных дочерних элементов (будут отобраны все нечетные дочерние элементы в блоке div на текущей странице)

$( " div:nth-child(2) " ) // выбор дочерних элементов с заданным индексом (будет отобран третий дочерний элемент в блоке div на текущей странице)

$( " div:nth-child(2n+1) " ) // выбор дочерних элементов по формуле (будут отобраны все нечетные дочерние элементы в блоке div на текущей странице)

*Фильтры форм*

$( " :enabled " ) // выбор всех активных элементов форм

$( " :disabled " ) // выбор всех неактивных элементов форм

$( " :checked " ) // выбор всех отмеченных элементов форм (выбранные радиокнопки и чекбоксы)

$( " :selected " ) // выбор всех выбранных элементов форм (выбранный option элемент в select)

**Методы манипуляции элементами**

Самым первым действием в jQuery является отбор нужных для дальнейшей работы элементов страницы, в результате получаем массив в виде объекта jQuery. Следующим шагом является применение методов, позволяющих работать с этим массивом. С помощью этих методов можно пополнять и уменьшать набор выбранных объектов, обходить их поэлементно, или создавать новые наборы, на основе уже существующего.

*Фильтрация отобранных элементов*

.is() //вернёт значение true, если среди отобранных ранее элементов присутствует хотя бы один, указанный в параметре и false - если такого элемента нет.

.not() // исключит из отобранных ранее элементы, которые указаны в параметре.

.eq(index) // выберет элемент из отобранных ранее, с указанным в параметре индексом.

.slice(startIndex, endIndex) // выберет элементы из ранее отобранных, индекс которых находится в диапазоне, указанном в параметре (отсчет начинается с 0). При указании отрицательных значений в параметре, отсчет начинается с 1. Второй параметр не обязателен.

.filter() // выберет элемент из отобранных ранее, которые соответствуют указанному в параметре условию.

.filter(fn) // выберет элементы из отобранных ранее, которые соответствуют значению, возвращенному из функции (при условии trye), указанной в параметре.

*Относительный поиск элементов*

.siblings() //возвращает элементы, которые являются соседними для отобранных ранее объектов и имеют общего родителя.

.parent() //возвращает элементы, которые являются прямыми родителями для отобранных ранее объектов.

.parents() //возвращает все родительские элементы для отобранных ранее объектов (независимо от уровня вложенности.

.children() //возвращает дочерние элементы каждого из отобранных ранее объектов

.next() //возвращает элемент, который расположен следующим за объектом из отобранных ранее.

.nextAll() ///возвращает все элементы, который расположены следующими за объектом из отобранных ранее.

.prev() //возвращает один элемент, который расположен перед объектом из отобранных ранее.

.prevAll() //возвращает все элементы, который расположены перед объектом из отобранных ранее.

*Работа с атрибутами элементов*

.attr(name) //Вернет значение атрибута, указанного в параметре метода, для первого объекта из отобранного набора элементов.

.attr(name, value) //Присвоит значение (value) атрибуту, указанному в параметре метода (name), для объектов из отобранного набора.

.attr(name, fn) //Присвоит значение атрибуту, указанному в параметре метода (name), которое возвращает пользовательская функция (fn), для всех объектов из отобранного набора.

.attr(settings) //Объектам из отобранного набора, присвоит атрибуты с соответствующими значениями, перечень которых указан в параметре метода.

.removeAttr(name) //Удалит атрибут, указанный в параметре метода, у всех отобранных ранее объектов, обладающих этим атрибутом.

*Работа с классами элементов*

.hasClass(name) //проверяет наличие элемента с классом, имеющим значение, указанное в параметре метода. Вернет true, если в выбранном наборе присутствует хотя бы один элемент у которого атрибут class имеет значение, указанное в параметре метода. В противном случае вернет false.

.addClass(name) //добавит к атрибуту class отобранных элементов значение (name), указанное в параметре метода.

.removeClass(name) //удалит значение (name) атрибута class, указанное в параметре метода, у всех элементов в выбранном наборе, которым присвоен атрибут class с указанным значением.

.toggleClass(name) //добавит атрибут class со значением (name), указанным в параметре метода, ко всем элементам в выбранном наборе, у которых отсутствует это значение, и удалит значение (name) атрибута class, указанное в параметре метода, у всех элементов в выбранном наборе, которым присвоен атрибут class с данным значением.

.toggleClass(name, switch) //Добавит сласс, указанный в первом параметре (name) метода, если второй параметр (функция switch) возвращает значение true, или удалит указанный класс, если функция switch вернет false.

*Работа с атрибутом value*

.val() //вернет значение атрибута value для первого элемента input, содержащегося в отобранном наборе. Для элемента select с возможностью множественного выбора, вернет массив значений (option), которые выбраны в первом элементе.

.val(string) //установит значение value, указанное в параметре (string) метода для всех отобранных элементов input

.val(array) //принимает в качестве параметра строковый массив со значениями для радиокнопок, чекбоксов и select

*Методы для работы с контентом*

.text() //возвращает текстовое содержимое всех элементов в отобранном наборе.

.text(string) //изменит текстовое содержимое элементов в отобранном наборе на строку, переданную в параметре (string) метода.

.html() //возвращает html содержимое первого элемента в отобранном наборе.

.html(string) //добавит html содержимое, переданное в параметре (string) метода, для всех элементов в отобранном наборе

*Методы для вставки контента*

.append(content) //вставляет контент, переданный в параметре (content) метода, внутрь каждого элемента из отобранного набора за уже существующим содержимым элемента.

.appendTo(selector) //перемещает элементы из отобранного набора внутрь элемента, указанного в параметре (selector) метода, в конец содержимого элемента.

.prepend(content) //вставляет контент, переданный в параметре (content) метода, внутрь каждого элемента из отобранного набора перед уже существующим содержимым элемента.

.prependTo(selector) //перемещает элементы из отобранного набора внутрь элемента, указанного в параметре (selector) метода, в начало содержимого элемента.

.before(content) //вставляет контент, переданный в параметре (content) метода, перед каждым элементом из отобранного набора.

.after(content) //вставляет контент, переданный в параметре (content) метода, после каждого элемента из отобранного набора.

.insertBefore(selector) //перемещает элементы из отобранного набора в позицию перед элементом, указанным в параметре (selector) метода.

.insertAfter(selector) //перемещает элементы из отобранного набора в позицию после элемента, указанного в параметре (selector) метода.

*Методы для обрамления контента*

.wrap(html) //оборачивает каждый из отобранных элементов в html, указанный в параметре.

.wrap(element) //оборачивает каждый отобранный элемент в DOM-элемент, указанный в параметре.

.wrapAll(html) //оборачивает все отобранные элементы в html, указанный в параметре.

.wrapAll(element) //оборачивает все отобранные элементы в DOM-элемент, указанный в параметре.

.wrapInner(html) //оборачивает содержимое каждого отобранного элемента в html, указанный в параметре.

.wrapInner(element) //оборачивает содержимое каждого отобранного элемента в DOM-элемент, указанный в параметре.

*Замена, удаление и клонирование элементов*

.replaceAll(selector) //заменит элемент, указанный в параметре метода, на элемент из отобранного набора.

.replaceWith(content) //заменит элементы из отобранного набора, строкой (включая html), переданной в параметре (content) метода.

.empty() //удаляет все содержимое (включая дочерние элементы) из объектов в отобранном наборе.

.remove(expression) //удаляет все элементы из отобранного набора. В качестве параметра может принимать дополнительные условия для удаления элементов.

.clone() //клонирует элементы из отобранного ранее набора. Если в параметре метода передать true, то будут скопированы и методы, примененные к клонируемому элементу.

**Методы анимации объектов**

Анимация элементов на странице - одна из самых мощных возможностей библиотеки jQuery. Начиная от простых возможностей - скрытия, появления, перемещения элементов, библиотека позволит создавать свои, более сложные эффекты, основанные на изменении CSS свойств. Эти изменения могут происходить плавно или мгновенно, замедляться, ускоряться или выполняться равномерно.

*Стандартная анимация*

.hide() //скрывает элементы из отобранного набора (без анимационных эффектов), если до применения метода элементы были видимы.

.hide(speed, fn) //скрывает элементы из отобранного набора, если до применения метода элементы были видимы. Принимает два необязательных параметра, скорость (speed) с которой будет спрятан элемент (указывается в миллисекундах, или с помощью ключевого слова) и callback функция, которая будет выполнена после исчезновения элемента.

.show() //показывает элементы из отобранного набора (без анимационных эффектов), если до применения метода элементы были скрыты.

.show(speed, fn) //показывает элементы из отобранного набора, если до применения метода элементы были скрыты. Принимает два необязательных параметра, скорость (speed) с которой будет показан элемент (указывается в миллисекундах, или с помощью ключевого слова) и callback функция, которая будет выполнена после того, как элемент станет видимым.

.toggle() //скрывает элементы из отобранного набора (без анимационных эффектов), если до применения метода элементы были видимы и показывает элементы из отобранного набора (без анимационных эффектов), если до применения метода элементы были скрыты (может использоваться как обработчик события onClick и применяет методы hide и show)

.toggle(switch) //Принимает в качестве параметра (switch) булево значение true или false. Вызывается непосредственно для изменения состояния выбранных элементов. Если параметр равен true, то элемент будет показан. Если параметр равен false, то элемент будет спрятан.

.toggle(speed, fn) //скрывает элементы из отобранного набора, если до применения метода элементы были видимы и показывает элементы из отобранного набора, если до применения метода элементы были скрыты. Принимает два необязательных параметра, скорость (speed) с которой будет показан/скрыт элемент (указывается в миллисекундах, или с помощью ключевого слова) и callback функция, которая будет выполнена после того, как элемент станет скрытым/видимым.

.fadeIn(speed, fn) //показывает элементы из отобранного набора, изменяя их прозрачность, если до применения метода элементы были скрыты.

.fadeOut(speed, fn) //скрывает элементы из отобранного набора, изменяя их прозрачность, если до применения метода элементы были видимы.

.fadeTo(speed, opacity, fn) //изменяет прозрачность отобранных элементов до заданного в параметрах метода значения.

.slideUp(speed, fn) //скрывает элементы из отобранного набора, изменяя их высоту, если до применения метода элементы были видимы.

.slideDown(speed, fn) //показывает элементы из отобранного набора, изменяя их высоту, если до применения метода элементы были скрыты.

.slideToggle(speed, fn) //скрывает элементы из отобранного набора, если до применения метода элементы были видимы и показывает элементы из отобранного набора, если элементы были скрыты (может использоваться как обработчик события onClick и применяет методы slideDown и slideUp)

*Управление анимацией*

.animate() //выполняет анимацию, заданную пользователем, с элементами из отобранного набора.

.queue() //добавляет, или изменяет очередь функций, в зависимости от переданных в методе параметров.

.clearQueue() //очищает очередь функций, примененных к элементам из отобранного набора.

.dequeue() //начинает выполнение следующей в очереди функции, примененной к отобранным элементам, не дожидаясь окончания выполнения предыдущей.

.stop() //прерывает выполнение текущей анимации, примененной к отобранному набору элементов.

.delay() //устанавливает временной промежуток, переданный в параметре метода, перед началом выполнения следующей в очереди функции, примененной к отобранному набору элементов.

jQuery.fx.interval //устанавливает временной промежуток между кадрами анимации, примененной к отобранному набору элементов. Свойство является глобальным для любой анимации в библиотеке.

jQuery.fx.off //отключает все анимации, выполняемые при помощи библиотеки jQuery, если методу передается параметр true.

**Обработчики событий jQuery**

Обработчики событий - это функции, код которых исполняется только после совершения определенных действий.

Обработчики событий присутствовали и в JavaScript, но jQuery облегчает их использование и расширяет их функциональность.

Примеры действий, после которых выполняются обработчики:

* Курсор мыши наведен на элемент;
* Веб-страница или картинка полностью загружена;
* Изменено содержимое поля формы;
* HTML форма отправлена;
* Нажата клавиша на клавиатуре;
* Общий вид определения обработчиков jQuery:

$(селектор).обработчик\_события(function(){код\_обработчика\_события});

Обработчики событий jQuery в действии

Код обработчика mouseover будет выполнен, когда курсор мыши будет наведен на элемент.

Код обработчика mouseout будет выполнен, когда курсор мыши будет выведен из границ элемента.

Код обработчика click будет выполнен после одинарного щелчка мыши на элементе.

Код обработчика dblclick будет выполнен после двойного щелчка мыши на элементе.

Код обработчика focus() будет выполнен, когда элемент станет активным.

Код обработчика blur() будет выполнен, когда элемент перестанет быть активным.

Код обработчика change() будет выполнен, при изменении содержимого элемента.

С помощью jQuery Вы можете также вызывать обработчики событий привязанные к элементу.

Например: $('#test').blur() вызовет обработчик blur у элемента с id='test'.

Объект event

Объект event содержит информацию о произошедшем событии. Объект event создается для каждого произошедшего события, но для того, чтобы иметь возможность обращаться к его свойствам и методам его необходимо явно передать в обработчик события.

Синтаксис:

$(селектор).событие(function(event){код\_обработчика\_события});

После того, как объект event был передан обработчику события, Вы можете обращаться к его свойствам и методам.

Управление обработчиками событий

Метод jQuery one() позволяет создавать обработчики, которые могут быть вызваны лишь один раз.

Метод jQuery toggle() позволяет переключаться между различными обработчиками событий по щелчку мыши.

**Пример: Таймер**



Index.html

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="utf-8" />

<link rel="stylesheet" type="text/css" href="style/myStyle.css">

<title>Timer</title>

</head>

<body>

<div id="time"></div>

<script type="text/javascript" src="js/jquery-1.11.3.min.js"></script>

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

</body>

</html>

myScript.js

function curentTime() {

var timerId = setInterval(function() {

var a = new Date();

var dday = new Array("воскресенье", "понедельник", "вторник", "среда", "четверг", "пятница", "суббота");

var Str = a.getDate() + " " + dday[a.getDay()] + " " + a.getFullYear() + " год " + a.getHours() + ":" + a.getMinutes() + ":" + a.getSeconds();

$('#time').html(Str);

}, 1000);

}

curentTime();

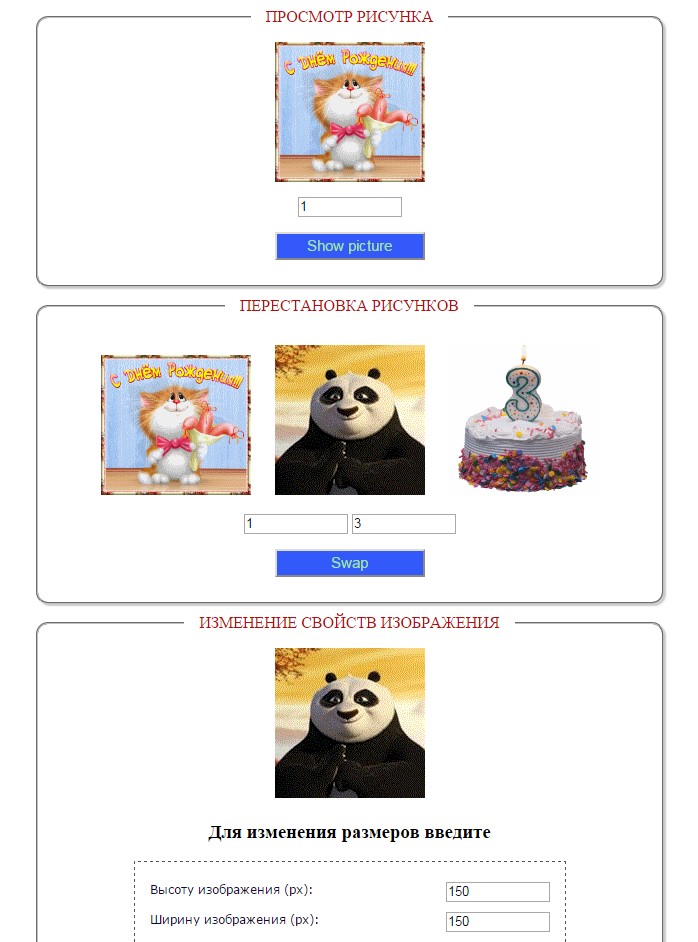
Создаем таймер timerId внутри функции curentTime , который будет вызваться каждую секунду. Значение таймера вставляем командой

$('#time').html(Str);

То есть в элемент с id = “time” мы вставляем с помощью метода jQuery html() значение сформированной нами строки Str

И в самом скрипте в конце выполняем функцию curentTime(); чтобы наш таймер запустился.

**Пример работы с изображениями**



<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="utf-8" />

<link rel="stylesheet" type="text/css" href="style/myStyle.css">

<title>Image</title>

</head>

<body>

<main class="content">

<fieldset>

<legend class="title">Просмотр рисунка</legend>

<img id="imgShow" class="first-img" src="img/1.gif" alt="pic" title="showpic">

<input type="number" min="1" max="3" value="1" id="showpic">

<button id="show">Show picture</button>

</fieldset>

<fieldset>

<legend class="title">Перестановка рисунков</legend>

<div class="wrap-img">

<img id="img1" src="img/1.gif" alt="first">

<img id="img2" src="img/2.gif" alt="second">

<img id="img3" src="img/3.gif" alt="thrid">

</div>

<input type="number" min="1" max="3" value="1" id="swapStart">

<input type="number" min="1" max="3" value="3" id="swapEnd">

<button id="swap" type="button">Swap</button>

</fieldset>

<fieldset>

<legend class="title">Изменение свойств изображения</legend>

<div class="wrap">

<img id="changeImg" src="img/2.gif" alt="Change Image" />

</div>

<h3>Для изменения размеров введите </h3>

<ul class="content-prop">

<li class="content-prop\_\_item clearfix">

<label for="imgHg" class="text"> Высоту изображения (px): </label>

<input type="number" id="imgHg" value="150" class="input">

</li>

<li class="content-prop\_\_item clearfix">

<label for="imgWd" class="text">Ширину изображения (px): </label>

<input type="number" id="imgWd" value="150" class="input">

</li>

</ul>

<h3> Для задания рамки введите</h3>

<ul class="content-prop">

<li class="content-prop\_\_item clearfix">

<label for="brdSz" class="text">Толщину рамки (px): </label>

<input type="number" id="brdSz" value="1" class="input">

</div>

<li class="content-prop\_\_item clearfix">

<label for="brdCl" class="text">Цвет рамки: </label>

<input type="color" id="brdCl" value="#0500ff" class="input">

</li>

<li class="content-prop\_\_item clearfix">

<label for="altImg" class="text">Алтернативный текст:</label>

<input type="text" id="altImg" value="avatar" class="input">

</li>

<li class="content-prop\_\_item clearfix">

<label for="algImg" class="text">Определите форматирование рисунка:</label>

<select id="algImg" class="input">

<option value="left">Влево</option>

<option value="right">Вправо</option>

<option value="center">По центру</option>

</select>

</li>

</ul>

<button id="imgDone" type="button">Done</button>

</fieldset>

</div>

</main>

<script src="js/jquery-1.11.3.min.js"></script>

<script src="js/myScript.js"></script>

</body>

</html>

**Index.html**

.content {

margin-left: auto;

margin-right: auto;

width: 1000px;

}

fieldset {

width: 600px;

border-radius: 12px;

box-shadow: 2px 2px 1px #ccc;

margin: 10px auto;

text-align: center;

}

.title {

text-align: center;

color: #b90b0b;

text-transform: uppercase;

font: bold;

padding: 0 15px;

margin-bottom: 10px;

}

.first-img {

display: block;

margin: 0 auto 15px;

}

input {

display: inline-block;

width: 100px;

}

button {

width: 150px;

font-size: 15px;

color: rgb(164, 227, 154);

background-color: #3359fa;

line-height: 1.5;

display: block;

margin: 15px auto;

outline: none;

}

.wrap-img {

margin: 15px auto;

}

#img1, #img2, #img3 {

margin: 0 10px;

}

.text {

display: block;

float:left;

}

.input {

display: block;

float:right;

}

.content-prop {

margin: 0;

padding: 15px;

width: 400px;

margin: 0 auto;

border: 1px dashed rgb(85, 82, 85);

}

.content-prop\_\_item {

list-style: none;

font-family: Verdana, serif;

color: rgb(14, 6, 47);

font-size: 12px;

padding: 5px 0;

}

/\* clearfix\*/

.clearfix:before,

.clearfix:after {

content: "";

display: table;

}

.clearfix:after {

clear: both;

}

**myStyle.css**

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

$('#imgShow').attr("src", "img/" + $("#showpic").val() + ".gif");

});

$('#swap').click(function() {

var $first = $('#img' + $('#swapStart').val());

var $second = $('#img' + $('#swapEnd').val());

var $temp = $first.attr("src");

$first.attr("src", $second.attr("src"));

$second.attr("src", $temp);

});

$('#imgDone').click(function() {

var color = $('#brdCl').val();

var brdWidth = $('#brdSz').val();

var alt = $('#altImg').val();

var wdImg = $('#imgWd').val();

var htImg = $('#imgHg').val();

var pos = $("#algImg").val();

$('#changeImg').width(wdImg)

.height(htImg)

.attr("alt", alt)

.css('border-width', brdWidth)

.css("border-color", color)

.css('border-style', 'solid');

$(".wrap").css('text-align', pos);

})

**myScript.js**

Так же не забудьте, что нам нужны три файла с расширением gif



1.gif



2.gif



3.gif

Остальные и эти примеры вы можете скачать по этой [ссылке](https://cloud.mail.ru/public/FBQK/zo4nf2oNh).

**Полезные ссылки**

[Учебник](http://anton.shevchuk.name/jquery-book/)

[Русская документация](http://jquery-docs.ru/)

[Справочник](http://jquery.page2page.ru/index.php5/%D0%97%D0%B0%D0%B3%D0%BB%D0%B0%D0%B2%D0%BD%D0%B0%D1%8F_%D1%81%D1%82%D1%80%D0%B0%D0%BD%D0%B8%D1%86%D0%B0)