

jQuery DOM



Добавление элементов

- Одним из способов создания элементов является передача строки, содержащей HTML-фрагмент, функции `$()`, которая выполнит синтаксический анализ строки и создаст соответствующие DOM-объекты.

.clone()

- Метод создает копии выбранных элементов страницы и возвращает их в виде объекта jQuery. Элементы копируются вместе со всеми содержащимися внутри них элементами (так называемое глубокое копирование).
- `.clone([withDataAndEvents])`
- `withDataAndEvents` — определяет, необходимо ли копировать данные и обработчики событий, установленные на копируемых элементах. По умолчанию, этот параметр равен `false`.
- `.clone([withDataAndEvents],[deepWithDataAndEvents])`
- `withDataAndEvents` — аналогично версии выше, за исключением одного: в jQuery-1.5.0 значение этого параметра по умолчанию было ошибочно изменено на `true`. В jQuery-1.5.1 значение по умолчанию было установлено обратно в `false`.
- `deepWithDataAndEvents` — определяет, нужно ли копировать данные и обработчики установленные на дочерних элементах клонируемого элемента. По умолчанию, принимает значение первого параметра.

.append() .appendTo()

- Функции добавляют содержимое в конец элементов. Имеется два варианта их использования:
- `elements.append(content)`, `content.appendTo(elements)`
- в конец элементов `elements` будет добавлен `content`, который может быть задан html-текстом, объектом jQuery или DOM объектом. Различия функций заключается только в порядке следования содержимого и элементов, в которые это содержимое должно быть помещено.
- `.append(function(index, value))`
- в конец выбранных элементов будет добавлен html-текст, который будет возвращен пользовательской функцией. Функция вызывается, отдельно для каждого из выбранных элементов. При вызове ей передаются следующие параметры: `index` — позиция элемента в наборе, `value` — текущий html-содержимое элемента.

.prepend() .prependTo()

- Функции добавляют содержимое в начало определенных элементов. Имеется два варианта использования функций:
- `elements.prepend(content)`, `content.prependTo(elements)`
- в начало элементов `elements` будет добавлен `content`, который может быть задан html-текстом, объектом jQuery или DOM объектом. Различия функций заключается только в порядке следования содержимого и элементов, в которые это содержимое должно быть помещено.
- `.prepend(function(index, value))`
- в начало выбранных элементов будет добавлен html-текст, который будет возвращен пользовательской функцией. Функция вызывается отдельно, для каждого из выбранных элементов. При вызове ей передаются следующие параметры: `index` — позиция элемента в наборе, `value` — текущий html-содержимое элемента.

.replaceWith() .replaceAll()

- С помощью этих функций можно заменять элементы страницы новыми элементами или уже существующими. Имеется два варианта использования функций:
- `elements.replaceWith(content)`, `content.replaceAll(elements)`:
- элементы `elements` будут заменены содержимым `content`, который может быть задан html-текстом, объектом jQuery или DOM объектом. Различия функций заключается только в порядке следования содержимого и элементов, которые будут заменены на это содержимое.
- `.replaceWith(function)`
- выбранные элементы будут заменены на содержимое, заданное html-текстом, который возвратит пользовательская функция. Функция вызывается отдельно, для каждого из выбранных элементов.

.before() .insertBefore()

- Функции помещают заданное содержимое перед определенными элементами страницы. Имеется два варианта использования функций:
- `elements.before(content)`, `content.insertBefore(elements)`
- перед элементами `elements` будет помещено содержимое `content`, которое может быть задано html-текстом, объектом jQuery или DOM объектом. Различия функций заключается только в порядке следования содержимого и элементов, перед которыми это содержимое должно быть помещено.
- `.before(function(index))`
- перед выбранными элементами будет добавлен html-текст, который будет возвращен пользовательской функцией. Функция вызывается отдельно, для каждого из выбранных элементов. При вызове ей передается один параметр: `index` — позиция элемента в наборе.

.after() .insertAfter()

- Функции вставляют заданное содержимое сразу после определенных элементов страницы. Имеется два варианта использования функций:
- `elements.after(content), content.insertAfter(elements)`
- сразу после элементов `elements` будет добавлено содержимое `content`, который может быть задан html-текстом, объектом jQuery или DOM объектом. Различия функций заключается только в порядке следования содержимого и элементов, после которых это содержимое должно быть вставлено.
- `.after(function(index))`
- после выбранных элементов будет добавлен html-текст, который будет возвращен пользовательской функцией. Функция вызывается отдельно, для каждого из выбранных элементов. При вызове ей передается один параметр: `index` — позиция элемента в наборе.

.wrap() .wrapAll()

- Функции помещают заданное содержимое вокруг выбранных элементов (как бы "обертывая" их). Имеется два варианта использования функций:
- .wrap(content), .wrapAll(content)
- выбранные элементы будут "обернуты" содержимым content, которое может быть задано html-текстом, объектом jQuery или DOM объектом. Различия методов заключается в том, что wrap оборачивает каждый выбранный элемент по отдельности, а wrapAll оборачивает все элементы сразу. Метод wrap включен в библиотеку начиная с версии 1.0, а wrapAll только с версии 1.2
- .wrap(function())
- выбранные элементы обертываются содержимым, которое будет возвращено в виде html-текста, пользовательской функцией. Функция вызывается отдельно, для каждого из выбранных элементов.

.wrapInner()

- "Обертывает" содержимое выбранных элементов заданными html-элементами. Таким образом получается, что все выбранные объекты "обертываются изнутри". Имеется два варианта использования этой функции:
 - .wrapInner(content)
 - "Обертывает изнутри" выбранные элементы заданным содержимым content, которое может быть задано html-текстом, объектом jQuery или DOM объектом.
 - .wrapInner(function())
 - "Обертывает изнутри" выбранные элементы заданным содержимым, которое будет возвращено в виде html-текста пользовательской функцией. Функция вызывается для каждого из выбранных элементов.

.remove() .detach()

- Методы для удаления элементов страницы.
- `.remove([selector]) .detach([selector])`
- Удаляют выбранные элементы на странице. В качестве параметра можно указать селектор и тогда удалены будут только те выбранные элементы, которые ему удовлетворяют.
- Различие двух рассматриваемых методов заключается в том, что при использовании `detach`, jQuery не удаляет информацию о элементе и поэтому он может быть восстановлен. Например:
- `var foo = jQuery('#foo');`
- `foo.detach(); //удаляем элемент`
- `//много-много кода`
- `foo.appendTo('body'); //вставляем элемент обратно на страницу (не обязательно в то же место, где он был)`

.unwrap()

- Удаляет родительские элементы у выбранных элементов, при этом, их содержимое останется на своих местах. Метод имеет один вариант использования:
- .unwrap()
- Метод не имеет параметров

.empty()

- Удаляет все содержимое у выбранных элементов (без удаления самих элементов). Метод имеет один вариант использования:
- .empty()
- Метод не имеет параметров.

.attr()

- Получает или устанавливает значение атрибута элементов набора jQuery в зависимости от заданного параметра.
- Варианты работы функции attr() в зависимости от переданных ей параметров:
 - attr (name) - получает значение атрибута name.
 - attr (name, value) - устанавливает значение атрибута name в value.

.prop()

- Получает или устанавливает значение свойства элементов набора jQuery в зависимости от заданного параметра.
- Варианты работы функции prop() в зависимости от переданных ей параметров:
- prop(name) - получает значение свойства name.
- prop (name, value) - устанавливает значение свойства name в value.

Отличие свойств и атрибутов

- Атрибуты – это то, что указано в HTML. Свойства – это вычисленные значения атрибутов в DOM