**Методы contains и compareDocumentPosition**

Если есть два элемента, то иногда бывает нужно понять, лежит ли один из них выше другого, то есть является ли его предком.

Обычные поисковые методы здесь не дают ответа, но есть два специальных. Они используются редко, но когда подобная задача встаёт, то знание метода может сэкономить много строк кода.

**[Метод contains для проверки на вложенность](http://learn.javascript.ru/compare-document-position" \l "metod-contains-dlya-proverki-na-vlozhennost)**

Синтаксис:

var result = parent.contains(child);

Возвращает true, если parent содержит child или parent == child.

**[Метод compareDocumentPosition для порядка узлов](http://learn.javascript.ru/compare-document-position" \l "metod-comparedocumentposition-dlya-poryadka-uzlov)**

Бывает, что у нас есть два элемента, к примеру, <li> в списке, и нужно понять, какой из них выше другого.

Метод compareDocumentPosition – более мощный, чем contains, он предоставляет одновременно информацию и о содержании и об относительном порядке элементов.

Синтаксис:

var result = nodeA.compareDocumentPosition(nodeB);

Возвращаемое значение – битовая маска (см. [Побитовые операторы](http://learn.javascript.ru/bitwise-operators)), биты в которой означают следующее:

| **Биты** | **Число** | **Значение** |
| --- | --- | --- |
| 000000 | 0 | nodeA и nodeB -- один и тот же узел |
| 000001 | 1 | Узлы в разных документах (или один из них не в документе) |
| 000010 | 2 | nodeB предшествует nodeA (в порядке обхода документа) |
| 000100 | 4 | nodeA предшествует nodeB |
| 001000 | 8 | nodeB содержит nodeA |
| 010000 | 16 | nodeA содержит nodeB |
| 100000 | 32 | Зарезервировано для браузера |

Понятие «предшествует» – означает не только «предыдущий сосед при общем родителе», но и имеет более общий смысл: "раньше встречается в порядке [прямого обхода](http://algolist.manual.ru/ds/walk.php) дерева документа.

Могут быть и сочетания битов. Примеры реальных значений:

<p>...</p>

<ul>

<li>1.1</li>

</ul>

<script>

var p = document.body.children[0];

var ul = document.body.children[1];

var li = ul.children[0];

// 1. <ul> находится после <p>

alert( ul.compareDocumentPosition(p) ); // 2 = 10

// 2. <p> находится до <ul>

alert( p.compareDocumentPosition(ul) ); // 4 = 100

// 3. <ul> родитель <li>

alert( ul.compareDocumentPosition(li) ); // 20 = 10100

// 4. <ul> потомок <body>

alert( ul.compareDocumentPosition(document.body) ); // 10 = 1010

</script>

Более подробно:

1. Узлы не вложены один в другой, поэтому стоит только бит «предшествования», отсюда 10.
2. То же самое, но обратный порядок узлов, поэтому 100.
3. Здесь стоят сразу два бита: 10100 означает, что ul одновременно содержит li и является его предшественником, то есть при прямом обходе дерева документа сначала встречается ul, а потом li.
4. Аналогично предыдущему, 1010 означает, что document.body содержит ul и предшествует ему.

**Перевод в двоичную систему**

Самый простой способ самостоятельно посмотреть, как число выглядит в 2-ной системе – вызвать для него toString(2), например:

var x = 20;

alert( x.toString(2) ); // "10100"

Или так:

alert( 20..toString(2) );

Здесь после 20 две точки, так как если одна, то JS подумает, что после неё десятичная часть – будет ошибка.

Проверить конкретное условие, например, "nodeA содержит nodeB", можно при помощи битовых операций, в данном случае: nodeA.compareDocumentPosition(nodeB) & 16, например:

<ul>

<li>1</li>

</ul>

<script>

var body = document.body;

var li = document.body.children[0].children[0];

if (body.compareDocumentPosition(li) & 16) {

alert( body + ' содержит ' + li );

}

</script>

Более подробно о битовых масках: [Побитовые операторы](http://learn.javascript.ru/bitwise-operators).

**[Поддержка в IE8-](http://learn.javascript.ru/compare-document-position" \l "podderzhka-v-ie8)**

В IE8- поддерживаются свои, нестандартные, метод и свойство:

[**nodeA.contains(nodeB)**](http://msdn.microsoft.com/en-us/library/ms536377.aspx)

Результат: true, если nodeA содержит nodeB, а также в том случае, если nodeA == nodeB.

[**node.sourceIndex**](http://msdn.microsoft.com/en-us/library/ms534635.aspx)

Номер элемента node в порядке прямого обхода дерева. Только для узлов-элементов.

На их основе можно написать полифилл для compareDocumentPosition:

// код с http://compatibility.shwups-cms.ch/en/polyfills/?&id=82

(function() {

var el = document.documentElement;

if (!el.compareDocumentPosition && el.sourceIndex !== undefined) {

Element.prototype.compareDocumentPosition = function(other) {

return (this != other && this.contains(other) && 16) +

(this != other && other.contains(this) && 8) +

(this.sourceIndex >= 0 && other.sourceIndex >= 0 ?

(this.sourceIndex < other.sourceIndex && 4) +

(this.sourceIndex > other.sourceIndex && 2) : 1

) + 0;

}

}

}());

С этим полифиллом метод доступен для элементов во всех браузерах.

**[Итого](http://learn.javascript.ru/compare-document-position" \l "itogo)**

* Для проверки, является ли один узел предком другого, достаточно метода nodeA.contains(nodeB).
* Для расширенной проверки на предшествование есть метод compareDocumentPosition.
* Для IE8 нужен полифилл для compareDocumentPosition.