**Стили, getComputedStyle**

Эта глава – о свойствах стиля, получении о них информации и изменении при помощи JavaScript.

Перед прочтением убедитесь, что хорошо знакомы с [блочной моделью CSS](http://www.w3.org/TR/CSS2/box.html) и понимаете, что такое padding, margin, border.

**[Стили элемента: свойство style](http://learn.javascript.ru/styles-and-classes" \l "stili-elementa-svoystvo-style)**

Свойство element.style возвращает объект, который дает доступ к стилю элемента на чтение и запись.

С его помощью можно изменять большинство CSS-свойств, например element.style.width="100px" работает так, как будто у элемента в атрибуте прописано style="width:100px".

**Единицы измерения обязательны в style**

Об этом иногда забывают, но в style так же, как и в CSS, нужно указывать единицы измерения, например px.

Ни в коем случае не просто elem.style.width = 100 – работать не будет.

**Для свойств, названия которых состоят из нескольких слов, используется вотТакаяЗапись:**

background-color => elem.style.backgroundColor

z N -index => elem.style.zIndex

border-left-width => elem.style.borderLeftWidth

Пример использования style:

document.body.style.backgroundColor = prompt('background color?', 'green');

**style.cssFloat вместо style.float**

Исключением является свойство float. В старом стандарте JavaScript слово "float" было зарезервировано и недоступно для использования в качестве свойства объекта. Поэтому используется не elem.style.float, а elem.style.cssFloat.

**Свойства с префиксами**

Специфические свойства браузеров, типа -moz-border-radius, -webkit-border-radius, записываются следующим способом:

button.style.MozBorderRadius = '5px';

button.style.WebkitBorderRadius = '5px';

То есть, каждый дефис даёт большую букву.

**Чтобы сбросить поставленный стиль, присваивают в style пустую строку: elem.style.width="".**

При сбросе свойства style стиль будет взят из CSS.

Например, для того, чтобы спрятать элемент, можно присвоить: elem.style.display = "none".

А вот чтобы показать его обратно – не обязательно явно указывать другой display, наподобие elem.style.display = "block". Можно просто снять поставленный стиль: elem.style.display = "".

// если запустить этот код, то <body> "мигнёт"

document.body.style.display = "none";

setTimeout(function() {

document.body.style.display = "";

}, 1000);

**Стиль в style находится в формате браузера, а не в том, в котором его присвоили.**

Например:

<body>

<script>

document.body.style.margin = '20px';

alert( document.body.style.marginTop ); // 20px!

document.body.style.color = '#abc';

alert( document.body.style.color ); // rgb(170, 187, 204)

</script>

</body>

Обратите внимание на то, как браузер «распаковал» свойство style.margin, предоставив для чтения style.marginTop. То же самое произойдет и для border, background и т.д.

**Свойство style мы используем лишь там, где не работают классы**

В большинстве случаев внешний вид элементов задаётся классами. А JavaScript добавляет или удаляет их. Такой код красив и гибок, дизайн можно легко изменять.

Свойство style нужно использовать лишь там, где классы не подходят, например если точное значение цвета/отступа/высоты вычисляется в JavaScript.

**[Строка стилей style.cssText](http://learn.javascript.ru/styles-and-classes" \l "stroka-stiley-style-csstext)**

Свойство style является специальным объектом, ему нельзя присваивать строку.

Запись div.style="color:blue" работать не будет. Но как же, всё-таки, поставить свойство стиля, если хочется задать его строкой?

Можно попробовать использовать атрибут: elem.setAttribute("style", ...), но самым правильным и, главное, кросс-браузерным (с учётом старых IE) решением такой задачи будет использование свойства style.cssText.

**Свойство style.cssText позволяет поставить стиль целиком в виде строки.**

Например:

<div>Button</div>

<script>

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

div.style.cssText="color: red !important; \

background-color: yellow; \

width: 100px; \

text-align: center; \

blabla: 5; \

";

alert(div.style.cssText);

</script>

Браузер разбирает строку style.cssText и применяет известные ему свойства. Неизвестные, наподобие blabla, большинство браузеров просто проигнорируют.

**При установке style.cssText все предыдущие свойства style удаляются.**

Итак, style.cssText осуществляет полную перезапись style. Если же нужно заменить какое-то конкретно свойство стиля, то обращаются именно к нему: style.color, style.width и т.п, чтобы не затереть что-то важное по ошибке.

Свойство style.cssText используют, например, для новосозданных элементов, когда старых стилей точно нет.

**[Чтение стиля из style](http://learn.javascript.ru/styles-and-classes" \l "chtenie-stilya-iz-style)**

Записать в стиль очень просто. А как прочитать?

Например, мы хотим узнать размер, отступы элемента, его цвет… Как это сделать?

**Свойство style содержит лишь тот стиль, который указан в атрибуте элемента, без учёта каскада CSS.**

Вот так style уже ничего не увидит:

<head>

<style> body { color: red; margin: 5px } </style>

</head>

<body>

Красный текст

<script>

alert(document.body.style.color); //в большинстве браузеров

alert(document.body.style.marginTop); // ничего не выведет

</script>

</body>

**[Полный стиль из getComputedStyle](http://learn.javascript.ru/styles-and-classes" \l "polnyy-stil-iz-getcomputedstyle)**

Итак, свойство style дает доступ только к той информации, которая хранится в elem.style.

Он не скажет ничего об отступе, если он появился в результате наложения CSS или встроенных стилей браузера:

А если мы хотим, например, сделать анимацию и плавно увеличивать marginTop от текущего значения? Как нам сделать это? Ведь для начала нам надо это текущее значение получить.

**Для того, чтобы получить текущее используемое значение свойства, используется метод window.getComputedStyle, описанный в стандарте**[**DOM Level 2**](http://www.w3.org/TR/2000/REC-DOM-Level-2-Style-20001113/css.html)**.**

Его синтаксис таков:

getComputedStyle(element[, pseudo])

**element**

Элемент, значения для которого нужно получить

**pseudo**

Указывается, если нужен стиль псевдо-элемента, например "::before". Пустая строка или отсутствие аргумента означают сам элемент.

Поддерживается всеми браузерами, кроме IE8-. Следующий код будет работать во всех не-IE браузерах и в IE9+:

<style>

body {

margin: 10px

}

</style>

<body>

<script>

var computedStyle = getComputedStyle(document.body);

alert( computedStyle.marginTop ); // выведет отступ в пикселях

alert( computedStyle.color ); // выведет цвет

</script>

</body>

**Вычисленное (computed) и окончательное (resolved) значения**

В CSS есть две концепции:

1. *Вычисленное* (computed) значение – это то, которое получено после применения всех правил CSS и CSS-наследования. Например, width: auto или font-size: 125%.
2. *Окончательное* ([resolved](http://dev.w3.org/csswg/cssom/" \l "resolved-values)) значение – непосредственно применяемое к элементу. При этом все размеры приводятся к пикселям, например width: 212px или font-size: 16px. В некоторых браузерах пиксели могут быть дробными.

Когда-то getComputedStyle задумывалось для возврата вычисленного значения, но со временем оказалось, что окончательное гораздо удобнее.

Поэтому сейчас в целом все значения возвращаются именно окончательные, кроме некоторых небольших глюков в браузерах, которые постепенно вычищаются.

**getComputedStyle требует полное свойство!**

Для правильного получения значения нужно указать точное свойство. Например: paddingLeft, marginTop, borderLeftWidth.

**При обращении к сокращенному: padding, margin, border – правильный результат не гарантируется.**

Действительно, допустим свойства paddingLeft/paddingTop взяты из разных классов CSS. Браузер не обязан объединять их в одно свойство padding. Иногда, в простейших случаях, когда свойство задано сразу целиком, getComputedStyle сработает для сокращённого свойства, но не во всех браузерах.

Например, некоторые браузеры (Chrome) выведут 10px в документе ниже, а некоторые (Firefox) – нет:

<style>

body {

margin: 10px;

}

</style>

<script>

var style = getComputedStyle(document.body);

alert( style.margin ); // в Firefox пустая строка

</script>

**Стили посещенных ссылок – тайна!**

У посещенных ссылок может быть другой цвет, фон, чем у обычных. Это можно поставить в CSS с помощью псевдокласса :visited.

Но getComputedStyle не дает доступ к этой информации, чтобы произвольная страница не могла определить, посещал ли пользователь ту или иную ссылку.

Кроме того, большинство браузеров запрещают применять к :visited CSS-стили, которые могут изменить геометрию элемента, чтобы даже окольным путем нельзя было это понять. В целях безопасности.

**[currentStyle для IE8-](http://learn.javascript.ru/styles-and-classes" \l "currentstyle-dlya-ie8)**

В IE8- нет getComputedStyle, но у элементов есть свойство [currentStyle](http://msdn.microsoft.com/en-us/library/ms536497.aspx), которое возвращает вычисленное (computed) значение: уже с учётом CSS-каскада, но не всегда в окончательном формате.

Чтобы код работал и в старых и новых браузерах, обычно пишут кросс-браузерный код, наподобие такого:

function getStyle(elem) {

return window.getComputedStyle ? getComputedStyle(elem, "") : elem.currentStyle;

}

Если вы откроете такой документ в IE8-, то размеры будут в процентах, а в современных браузерах – в пикселях.

<style>

body {

margin: 10%

}

</style>

<body>

<script>

var elem = document.body;

function getStyle(elem) {

return window.getComputedStyle ? getComputedStyle(elem, "") : elem.currentStyle;

}

var marginTop = getStyle(elem).marginTop;

alert( marginTop ); // IE8-: 10%, иначе пиксели

</script>

</body>

**IE8-: перевод pt,em,% из currentStyle в пиксели**

Эта информация – дополнительная, она не обязательна для освоения.

В IE для того, чтобы получить из процентов реальное значение в пикселях существует метод «runtimeStyle+pixel», [описанный Дином Эдвардсом](http://erik.eae.net/archives/2007/07/27/18.54.15/#comment-102291).

Он основан на свойствах runtimeStyle и pixelLeft, работающих только в IE.

В [песочнице](https://plnkr.co/edit/ZI2dH1pGoBRHNvgpi4YC?p=preview) вы можете найти функцию getIEComputedStyle(elem, prop), которая получает значение в пикселях для свойства prop, используя elem.currentStyle и метод Дина Эдвардса, и пример её применения.

Если вам интересно, как он работает, ознакомьтесь со свойствами с [runtimeStyle](http://msdn.microsoft.com/en-us/library/ms535889(v=vs.85).aspx) и [pixelLeft](http://msdn.microsoft.com/en-us/library/ms531129%28VS.85%29.aspx) в MSDN и раскройте код.

Конечно, это актуально только для IE8- и полифиллов.

**[Итого](http://learn.javascript.ru/styles-and-classes" \l "itogo)**

Все DOM-элементы предоставляют следующие свойства.

* Свойство style – это объект, в котором CSS-свойства пишутся вотТакВот. Чтение и изменение его свойств – это, по сути, работа с компонентами атрибута style.
* style.cssText – строка стилей для чтения или записи. Аналог полного атрибута style.
* Свойство currentStyle(IE8-) и метод getComputedStyle (IE9+, стандарт) позволяют получить реальное, применённое сейчас к элементу свойство стиля с учётом CSS-каскада и браузерных стилей по умолчанию.

При этом currentStyle возвращает значение из CSS, до окончательных вычислений, а getComputedStyle – окончательное, непосредственно применённое к элементу (как правило).

Более полная информация о свойстве style, включающая другие, реже используемые методы работы с ним, доступна [в документации](https://developer.mozilla.org/en-US/docs/DOM/CSSStyleDeclaration).