**Приём проектирования "поведение"**

Шаблон проектирования «поведение» (behavior) позволяет задавать хитрые обработчики на элементе *декларативно*, установкой специальных HTML-атрибутов и классов.

**[Описание](http://learn.javascript.ru/behavior" \l "opisanie)**

Приём проектирования «поведение» состоит из двух частей:

1. Элементу ставится атрибут, описывающий его поведение.
2. При помощи делегирования ставится обработчик на документ, который ловит все клики и, если элемент имеет нужный атрибут, производит нужное действие.

**[Пример](http://learn.javascript.ru/behavior" \l "primer)**

Например, добавим «поведение», которое всем элементам, у которых стоит атрибут data-counter, будет при клике увеличивать значение на 1:

Счётчик:

<button data-counter>1</button>

Ещё счётчик:

<button data-counter>2</button>

<script>

document.onclick = function(event) {

if (!event.target.hasAttribute('data-counter')) return;

var counter = event.target;

counter.innerHTML++;

};

</script>

Если запустить HTML-код выше, то при клике на каждую кнопку – её значение будет увеличиваться.

Конечно, нам важны не счётчики, а общий подход, который они демонстрируют.

Элементов data-counter может быть сколько угодно. Новые могут добавляться в HTML в любой момент. При помощи делегирования мы, фактически, добавили новый «псевдо-стандартный» атрибут в HTML, который добавляет элементу новую возможность («поведение»).

**[Ещё пример](http://learn.javascript.ru/behavior" \l "eschyo-primer)**

Добавим ещё поведение.

Сделаем так, что при клике на элемент с атрибутом data-toggle-id будет скрываться/показываться элемент с заданным id:

<button data-toggle-id="subscribe-mail">

Показать форму подписки

</button>

<form id="subscribe-mail" hidden>

Ваша почта: <input type="email">

</form>

<script>

document.onclick = function(event) {

var target = event.target;

var id = target.getAttribute('data-toggle-id');

if (!id) return;

var elem = document.getElementById(id);

elem.hidden = !elem.hidden;

};

</script>

Ещё раз заметим, что мы сделали. Теперь для того, чтобы добавить скрытие-раскрытие любому элементу – даже не надо знать JavaScript, можно просто написать атрибут data-toggle-id.

Это бывает очень удобно – не нужно писать JavaScript-код для каждого элемента, который должен служить такой кнопкой. Просто используем поведение.

Обратите внимание: обработчик поставлен на document, клик на любом элементе страницы пройдёт через него, так что поведение определено глобально.

**Не только атрибут**

Для своих целей мы можем использовать в HTML любые атрибуты, но стандарт рекомендует для своих целей называть атрибуты data-\*.

В обработчике document.onclick мы могли бы проверять не атрибут, а класс или что-то ещё, но с атрибутом – проще и понятнее всего.

Также для добавления обработчиков на document рекомендуется использовать addEventListener, чтобы можно было добавить более одного обработчика для типа события.

**[Итого](http://learn.javascript.ru/behavior" \l "itogo)**

Шаблон «поведение» удобен тем, что сколь угодно сложное JavaScript-поведение можно «навесить» на элемент одним лишь атрибутом. А можно – несколькими атрибутами на связанных элементах.

Здесь мы рассмотрели базовый пример, который можно как угодно модифицировать и масштабировать. Важно не переусердствовать.

Приём разработки «поведение» рекомендуется использовать для расширения возможностей разметки, как альтернативу мини-фрагментам JavaScript.