

Раздел

[Формы, элементы управления](#)

Навигация по уроку

События focus/blur

Методы focus/blur

Включаем фокусировку на любом элементе: tabindex


События focusin/focusout

Итого

Задачи (3)

Комментарии

Поделиться

[Редактировать на GitHub](#)[🏠 → Браузер: документ, события, интерфейсы](#)
[→ Формы, элементы управления](#) 6-го сентября 2019

Фокусировка: focus/blur

Элемент получает фокус, когда пользователь кликает по нему или использует клавишу `Tab`. Также существует HTML-атрибут `autofocus`, который устанавливает фокус на элемент, когда страница загружается. Есть и другие способы получения фокуса, о них – далее.

Фокусировка обычно означает: «приготовься к вводу данных на этом элементе», это хороший момент, чтобы инициализировать или загрузить что-нибудь.

Момент потери фокуса («blur») может быть важнее. Это момент, когда пользователь кликает куда-то ещё или нажимает `Tab`, чтобы переключиться на следующее поле формы. Есть другие причины потери фокуса, о них – далее.

Потеря фокуса обычно означает «данные введены», и мы можем выполнить проверку введённых данных или даже отправить эти данные на сервер и так далее.

В работе с событиями фокусировки есть важные особенности. Мы постараемся разобрать их далее.

События focus/blur

Событие `focus` вызывается в момент фокусировки, а `blur` – когда элемент теряет фокус.

Используем их для валидации(проверки) введённых данных.

В примере ниже:

- Обработчик `blur` проверяет, введён ли email, и если нет – показывает ошибку.
- Обработчик `focus` скрывает это сообщение об ошибке (в момент потери фокуса проверка повторится):

```
1 <style>
2   .invalid { border-color: red; }
3   #error { color: red }
4 </style>
5
6 Ваш email: <input type="email" id="input">
7
8 <div id="error"></div>
9
10 <script>
11   input.onblur = function() {
12     if (!input.value.includes('@')) { // не email
13       input.classList.add('invalid');
14       error.innerHTML = 'Пожалуйста, введите правильный е
15     }
16   };
17
18   input.onfocus = function() {
19     if (this.classList.contains('invalid')) {
20       // удаляем индикатор ошибки, т.к. пользователь хоче
21       this.classList.remove('invalid');
22       error.innerHTML = "";
23     }
24   };
25 </script>
```

Ваш email:

Раздел

Формы, элементы управления

Навигация по уроку

События focus/blur

Методы focus/blur

Включаем фокусировку на любом элементе: tabindex

События focusin/focusout

Итого

Задачи (3)

Комментарии

Поделиться



Редактировать на GitHub



Современный HTML позволяет делать валидацию с помощью атрибутов `required`, `pattern` и т.д. Иногда – это всё, что нам нужно. JavaScript можно использовать, когда мы хотим больше гибкости. А ещё мы могли бы отправлять изменённое значение на сервер, если оно правильное.

Методы focus/blur

Методы `elem.focus()` и `elem.blur()` устанавливают/снимают фокус.

Например, запретим посетителю переключаться с поля ввода, если введённое значение не прошло валидацию:

```
1 <style>
2   .error {
3     background: red;
4   }
5 </style>
6
7 Ваш email: <input type="email" id="input">
8 <input type="text" style="width:280px" placeholder="вве,
9
10 <script>
11   input.onblur = function() {
12     if (!this.value.includes('@')) { // не email
13       // показать ошибку
14       this.classList.add("error");
15       // ...и вернуть фокус обратно
16       input.focus();
17     } else {
18       this.classList.remove("error");
19     }
20   };
21 </script>
```



Ваш email: введите неверный email и кликните сюда



Это работает во всех браузерах, кроме Firefox ([bug](#)).

Если мы что-нибудь введём и нажмём `Tab` или кликнем в другое место, тогда `onblur` вернёт фокус обратно.

Отметим, что мы не можем «отменить потерю фокуса», вызвав `event.preventDefault()` в обработчике `onblur` потому, что `onblur` срабатывает *после* потери фокуса элементом.

⚠ Потеря фокуса, вызванная JavaScript

Потеря фокуса может произойти по множеству причин.

Одна из них – когда посетитель кликает куда-то ещё. Но и JavaScript может быть причиной, например:

- `alert` переводит фокус на себя – элемент теряет фокус (событие `blur`), а когда `alert` закрывается – элемент получает фокус обратно (событие `focus`).
- Если элемент удалить из DOM, фокус также будет потерян. Если элемент добавить обратно, то фокус не вернётся.

Из-за этих особенностей обработчики `focus/blur` могут сработать тогда, когда это не требуется.

Используя эти события, нужно быть осторожным. Если мы хотим отследить потерю фокуса, которую инициировал пользователь, тогда нам следует избегать её самим.

Раздел

Формы, элементы управления

Навигация по уроку

События focus/blur

Методы focus/blur

Включаем фокусировку на любом элементе: tabindex

События focusin/focusout

Итого

Задачи (3)

Комментарии

Поделиться



Редактировать на GitHub



Включаем фокусировку на любом элементе: tabindex

Многие элементы по умолчанию не поддерживают фокусировку.

Какие именно – зависит от браузера, но одно всегда верно: поддержка focus/blur гарантирована для элементов, с которыми посетитель может взаимодействовать: <button>, <input>, <select>, <a> и т.д.

С другой стороны, элементы форматирования <div>, , <table> – по умолчанию не могут получить фокус. Метод elem.focus() не работает для них, и события focus/blur никогда не срабатывают.

Это можно изменить HTML-атрибутом tabindex.

Любой элемент поддерживает фокусировку, если имеет tabindex. Значение этого атрибута – порядковый номер элемента, когда клавиша Tab (или что-то аналогичное) используется для переключения между элементами.

То есть: если у нас два элемента, первый имеет tabindex="1", а второй tabindex="2", то находясь в первом элементе и нажав Tab – мы переместимся во второй.

Порядок перебора таков: сначала идут элементы со значениями tabindex от 1 и выше, в порядке tabindex, а затем элементы без tabindex (например, обычный <input>).

При совпадающих tabindex элементы перебираются в том порядке, в котором идут в документе.

Есть два специальных значения:

- tabindex="0" ставит элемент в один ряд с элементами без tabindex. То есть, при переключении такие элементы будут после элементов с tabindex ≥ 1.

Обычно используется, чтобы включить фокусировку на элементе, но не менять порядок переключения. Чтобы элемент мог участвовать в форме наравне с обычными <input>.

- tabindex="-1" позволяет фокусироваться на элементе только программно. Клавиша Tab проигнорирует такой элемент, но метод elem.focus() будет действовать.

Например, список ниже. Кликните первый пункт в списке и нажмите Tab:

```
1 Кликните первый пункт в списке и нажмите Tab. Продолжай
2 <ul>
3   <li tabindex="1">Один</li>
4   <li tabindex="0">Ноль</li>
5   <li tabindex="2">Два</li>
6   <li tabindex="-1">Минус один</li>
7 </ul>
8
9 <style>
10   li { cursor: pointer; }
11   :focus { outline: 1px dashed green; }
12 </style>
```

Раздел

Формы, элементы управления

Навигация по уроку

События focus/blur

Методы focus/blur

Включаем фокусировку на любом элементе: tabindex

События focusin/focusout

Итого

Задачи (3)

Комментарии

Поделиться



Редактировать на GitHub



Кликните первый пункт в списке и нажмите Tab. Продолжайте следить за порядком. Обратите внимание, что много последовательных нажатий Tab могут вывести фокус из iframe с примером.

- Один
- Ноль
- Два
- Минус один

Порядок такой: 1 - 2 - 0. Обычно `` не поддерживает фокусировку, но `tabindex` включает её, а также события и стилизацию псевдоклассом `:focus`.

📘 Свойство `elem.tabIndex` тоже работает

Мы можем добавить `tabindex` из JavaScript, используя свойство `elem.tabIndex`. Это даст тот же эффект.

События focusin/focusout

События `focus` и `blur` не всплывают.

Например, мы не можем использовать `onfocus` на `<form>`, чтобы подсветить её:

```
1 <!-- добавить класс при фокусировке на форме -->
2 <form onfocus="this.className='focused'">
3   <input type="text" name="name" value="Имя">
4   <input type="text" name="surname" value="Фамилия">
5 </form>
6
7 <style> .focused { outline: 1px solid red; } </style>
```

Имя

Фамилия

Пример выше не работает, потому что когда пользователь перемещает фокус на `<input>`, событие `focus` срабатывает только на этом элементе. Это событие не всплывает. Следовательно, `form.onfocus` никогда не срабатывает.

У этой проблемы два решения.

Первое: забавная особенность – `focus/blur` не всплывают, но передаются вниз на фазе перехвата.

Это работает:

```
1 <form id="form">
2   <input type="text" name="name" value="Имя">
3   <input type="text" name="surname" value="Фамилия">
4 </form>
5
6 <style> .focused { outline: 1px solid red; } </style>
7
8 <script>
9
10
11
```

Раздел

Формы, элементы
управления

Навигация по уроку

События focus/blur

Методы focus/blur

Включаем фокусировку на
любом элементе: tabindex

События focusin/focusout

Итого

Задачи (3)

Комментарии

Поделиться



Редактировать на GitHub



```
12 // установить обработчик на фазе перехвата (последний
    form.addEventListener("focus", () => form.classList.a
    form.addEventListener("blur", () => form.classList.re
</script>
```

Имя	Фамилия
-----	---------

Второе решение: события focusin и focusout – такие же, как и focus/blur, но они всплывают.

Заметьте, что эти события должны использоваться с elem.addEventListener, но не с on<event>.

Второй рабочий вариант:

```
1 <form id="form">
2   <input type="text" name="name" value="Имя">
3   <input type="text" name="surname" value="Фамилия">
4 </form>
5
6 <style> .focused { outline: 1px solid red; } </style>
7
8 <script>
9   form.addEventListener("focusin", () => form.classList
10  form.addEventListener("focusout", () => form.classLis
11 </script>
```

Имя	Фамилия
-----	---------

Итого

События focus и blur срабатывают на фокусировке/потере фокуса элемента.

Их особенности:

- Они не всплывают. Но можно использовать фазу перехвата или focusin/focusout.
- Большинство элементов не поддерживают фокусировку по умолчанию. Используйте tabindex, чтобы сделать фокусируемым любой элемент.

Текущий элемент с фокусом можно получить из document.activeElement.

✓ Задачи

Редактируемый div

важность: 5

Создайте <div>, который превращается в <textarea>, если на него кликнуть.

<textarea> позволяет редактировать HTML в элементе <div>.

Когда пользователь нажимает `Enter` или переводит фокус, <textarea> превращается обратно в <div>, и его содержимое становится HTML-кодом в <div>.

[Демо в новом окне](#)

[Открыть песочницу для задачи.](#)

решение

Редактирование TD по клику

Раздел

Формы, элементы
управления

Навигация по уроку

События focus/blur

Методы focus/blur

Включаем фокусировку на
любом элементе: tabindex

События focusin/focusout

Итого

Задачи (3)

Комментарии

Поделиться



Редактировать на GitHub

важность: 5

Сделайте ячейки таблицы редактируемыми по клику.

- По клику – ячейка должна стать «редактируемой» (textarea появляется внутри), мы можем изменять HTML. Изменение размера ячейки должно быть отключено.
- Кнопки ОК и ОТМЕНА появляются ниже ячейки и, соответственно, завершают/отменяют редактирование.
- Только одну ячейку можно редактировать за один раз. Пока <td> в «режиме редактирования», клики по другим ячейкам игнорируются.
- Таблица может иметь множество ячеек. Используйте делегирование событий.

Демо:

Кликните на ячейку таблицы, чтобы редактировать её. Нажмите ОК или ОТМЕНА, когда закончите.

Квадрат *Bagua*: Направление, Элемент, Цвет, Значение

Северо-Запад
Металл
Серебро
Старейшины

Север
Вода
Синий
Перемены

Северо-Восток
Земля
Жёлтый
Направление

Запад
Металл
Золото
Молодость

Центр
Всё
Пурпурный
Гармония

Восток
Дерево
Синий
Будущее

Юго-Запад
Земля
Коричневый
Спокойствие

Юг
Огонь
Оранжевый
Слава

Юго-Восток
Дерево
Зеленый
Роман

Открыть песочницу для задачи.

решение

Мышь, управляемая клавиатурой

важность: 4

Установите фокус на мышь. Затем используйте клавиши со стрелками, чтобы её двигать:

[Демо в новом окне](#)

P.S. Не добавляйте обработчики никуда, кроме элемента #mouse . P.P.S. Не изменяйте HTML/CSS, подход должен быть общим и работать с любым элементом.

Открыть песочницу для задачи.

решение

Проводим [курсы по JavaScript и фреймворкам.](#)



Комментарии

перед тем как писать...

