hillel IT school

Lesson 20

План заняття

- Основні елементи керування;
- Checkboxes;
- Select;
- Form;

Навігація

Форми в документі є членами спеціальної колекції document.forms.

Коли у нас є форма, будь-який її елемент доступний в іменованій колекції form.elements.

У колекції може бути кілька елементів з однаковим ім’ям (name). Це типово для перемикачів (radio buttons) та чекбоксів (checkboxes).В такому випадку form.elements[name]

Зворотній доступ

Для будь-якого елемента форми, сама форма доступна у властивості element.form.

input та textarea

Ми можемо отримати доступ до значення через властивість

- Input.value
- Input.checked
- Використовуйте textarea.value замість textarea.innerHTML

select та option

Елемент <select> має 3 важливі властивості:

- select.options – набір піделементів <option>,
- select.value – значення поточного обраного елемента <option>,
- select.selectedIndex – номер поточного обраного елемента <option>.

Вони надають три різні способи встановлення значення для <select>:

- Знайти відповідний елемент <option> (наприклад, серед select.options) і встановити для його властивості option.selected значення true.
- Якщо ми знаємо нове значення: встановити нове значення для select.value.
- Якщо ми знаємо порядковий номер опції: встановити це число для select.selectedIndex.

На відміну від більшості інших елементів керування, <select> дозволяє вибрати декілька опцій одночасно, якщо він має атрибут multiple.

Специфікація

new Option

const option = new Option(text, value, defaultSelected, selected);

Цей синтаксис необов’язковий. Ми можемо використати document.createElement('option') і встановити атрибути вручну. Однак те саме можна зробити коротше, тому ось параметри:

- text – текст всередині опції,
- value – значення опції,
- defaultSelected – якщо true, то до опції буде додано HTML-атрибут selected,
- selected – якщо true, то опція буде обраною.

Елементи <option> мають такі властивості:

- option.selected - Вказує чи обрана опція.
- option.index - Номер опції серед інших в елементі <select>.
- option.text - Текстовий зміст опції (те, що бачить відвідувач).

focus/blur

Подія focus викликається в момент фокусування, а подія blur – коли елемент втрачає фокус.

Використаймо їх для валідації поля форми.

Методи elem.focus() та elem.blur() встановлюють/прибирають фокус на елементі.

tabindex

Багато елементів не підтримують фокусування за замовчуванням.

Їх список відрізняється у різних браузерах, але одне завжди вірно: підтримка focus/blur гарантована для елементів з якими користувач може взаємодіяти: <button>, <input>, <select>, <a> та інших.

Якщо елемент має атрибут tabindex, то на ньому можна фокусуватися. Значенням атрибуту є порядковий номер елементу, коли для переходу між елементами використовується клавіша Tab (або аналогічна).

Елементи з однаковим значенням tabindex отримують фокус у типовому порядку, так як вони розміщені в документі.

Є два спеціальні значення:

- tabindex="0" включає елемент у звичайний порядок елементів без tabindex. Тобто, коли ми перемикаємося між елементами, ті, що мають tabindex=0 йдуть після тих, що з tabindex ≥ 1. Зазвичай цей прийом використовують, щоб на елементі можна було сфокусуватися не змінюючи стандартний порядок перемикання. Щоб елемент став частиною форми на рівні з <input>.
- tabindex="-1" дозволяє лише програмне фокусування на елементі. Клавіша Tab ігнорує такі елементи, проте метод elem.focus() спрацьовує.

Делегування: focusin/focusout

Події focus та blur не спливають.

Наприклад, ми не можемо встановити onfocus на <form> щоб виділити її.

Існує два рішення.

1. цікава історична особливість: focus/blur не спливає, проте розповсюджується вниз на фазі захоплення.
2. є події focusin та focusout – такі ж самі як focus/blur, тільки вони спливають. Вони мають використовуватися з elem.addEventListener, а не з on<event>.

change

Подія change спрацьовує після закінчення зміни елемента.

Для текстового поля це означає, що подія відбувається, коли втрачається фокус.

Для інших елементів: select, input type=checkbox/radio подія запускається відразу після зміни значення

input

Подія input запускається щоразу після того, як користувач змінює значення. input запускається при будь-якій зміні значень, навіть тих, які не передбачають дії клавіатури: вставлення тексту за допомогою миші або використання розпізнавання мовлення для диктування тексту.

cut, copy, paste

Ці події відбуваються під час вирізання/копіювання/вставлення значення.

Дані зберігаюсть у буфері обміну системи.

Буфер обміну – це “глобальна” річ на рівні ОС. Користувач може перемикатися між різними програмами, копіювати/вставляти різні речі, і сторінка браузера не повинна бачити всього цього.

Тому більшість браузерів надають безперешкодний доступ для читання/запису до буфера обміну лише в рамках певних дій користувача, таких як копіювання/вставлення тощо.

Заборонено генерувати “користувацькі” події буфера обміну з dispatchEvent у всіх браузерах, крім Firefox. І навіть якщо нам вдасться відправити таку подію, у специфікації чітко зазначено, що такі “синтетичні” події не повинні надавати доступ до буфера обміну.

submit

Існує два основних способи надіслати форму:

1. клікнути <input type="submit">.
2. натиснути Enter у полі введення.Існує два основних способи надіслати форму.

Обидві дії призводять до події submit у формі. Обробник може перевірити дані, і якщо є помилки, показати їх і викликати event.preventDefault(), тоді форму не буде відправлено на сервер.

Коли форма надсилається за допомогою Enter у полі введення, запускається подія click на <input type="submit">.

Щоб надіслати форму на сервер вручну, ми можемо викликати form.submit().

Тоді подія submit не генерується. Передбачається, що якщо програміст викликає form.submit(), то сценарій вже здійснив всю пов’язану обробку.

Дякую за увагу