

Конспект «Формы. Погружение»

Простая кнопка

Самая простая кнопка создаётся с помощью тега `<input>` с типом `button`. При нажатии на такую кнопку никаких действий не происходит, а все необходимые действия обычно задаются при помощи JavaScript.

Пример записи:

```
<input type="button" value="Кнопка">
```

Сброс введенных значений

В HTML-формах есть специальная кнопка, которая сбрасывает введенные значения и возвращает изначально установленные. Это поле ввода с типом `reset`.

Пример использования:

```
<input type="reset" value="Сбросить">
```

Кнопка возвращает те значения, которые были установлены в полях формы по умолчанию.

Кнопка-изображение

В качестве кнопки отправки формы можно использовать изображение. Для этого у тега `input` нужно указать тип `image`.

У кнопки-изображения есть еще два атрибута:

- `src` задаёт адрес изображения;
- `alt` задаёт альтернативный текст, отображаемый в том случае, если изображение не загружено.

Кнопка-изображение работает аналогично кнопке `submit`, но на сервер дополнительно передаются координаты точки, по которой был произведен щелчок.

Альтернативный способ задания кнопок

Помимо тега `<input>` для добавления кнопок можно использовать тег `<button>`.
Внутри тега `<button>` можно размещать любые HTML-элементы, в том числе изображения. Например:

```
<button>Календарь </button>
```

По умолчанию значение атрибута `type` у тега `<button>` — `submit`.

Значениями атрибута `type` также могут быть `button` и `reset`.

Кнопка `button` со значением `type="reset"` сбрасывает значения полей формы к изначальным.

Значение `type="button"` избавит кнопку от всей изначальной функциональности.

Обязательные поля

Чтобы указать, что поле обязательно для заполнения, нужно добавить ему пустой атрибут `required`:

```
<input type="text" required>
```

При попытке отправить форму с незаполненными обязательными полями браузер выведет всплывающее предупреждение. Эта проверка работает на клиентской части и упрощает валидацию форм. Но всегда нужно проверять отправленные данные и на стороне сервера.

Поле выбора даты и времени

Для выбора даты из календаря существует новый тип поля ввода — `date`. При клике на данное поле в форме всплывает календарик.

Пример записи:

```
<input type="date">
```

Для указания времени существуют дополнительные «временные» типы полей, например, `time` для выбора времени.

```
<input type="time">
```

Также существуют следующие типы полей:

- `datetime-local` задаёт дату с указанием времени (без учета временной зоны);
- `week` задаёт порядковый номер недели в году и года;
- `month` задаёт месяц и год.

Если браузер не поддерживает указанный тип поля, то вместо него отображается обычное текстовое поле.

Список возможных значений

Для текстовых полей можно заранее определить список возможных значений, которые отображаются, когда вы начинаете вводить текст в поле. Для этого существует специальный тег `<datalist>`.

Пример использования:

```
<input type="text" list="browsers" name="browser">

<datalist id="browsers">

  <option value="Firefox"></option>

  <option value="Chrome"></option>

  <option value="Safari"></option>

</datalist>
```

Связывание текстового поля и списка осуществляется при помощи атрибута `list` у тега `input` — значение `list` должно быть таким же, как значение атрибута `id` у списка.

Если тег `input` имеет специфический тип, например `email` или другие, то в списке отображаются только корректные для данного типа значения.

Поле ввода числового значения

Для ввода числовых значений существует специальный тип поля ввода `number`. Рядом с полем браузер автоматически подставляет две стрелочки для увеличения и уменьшения числового значения.

Пример записи:

```
<input type="number">
```

Верхняя и нижняя границы числовых значений задаются при помощи вспомогательных атрибутов `min` и `max`. А атрибут `step` устанавливает величину шага изменения значения.

Поле поиска

Поле поиска задаётся при помощи тега `<input>` с типом `search`.

Пример записи:

```
<input type="search">
```

В некоторых браузерах внутри него появляется крестик для сброса введённого значения.

Автофокус

Для того, чтобы отметить поле, в которое по умолчанию нужно установить курсор используется пустой атрибут `autofocus`.

Пример записи:

```
<input type="text" autofocus>
```

Такой атрибут должен быть только один на странице.

Выбор из диапазона

Для выбора значения из диапазона подходит тип поля `range`.

Такое поле выглядит как шкала с ползунком и позволяет выбрать число из некоторого интервала значений.

Пример записи:

```
<input type="range" min="1" max="10">
```

Атрибуты `min` и `max` устанавливают нижнюю и верхнюю границу допустимых значений, а атрибут `step` устанавливает величину шага изменения значения.

Область для вывода результата

Тег `<output>` представляет собой область, куда выводятся какие-либо результаты вычислений, обычно полученные при помощи JavaScript.

Пример записи:

```
<output name="sum">[значение по умолчанию]</output>
```

Значение по умолчанию при этом можно не задавать, тогда область вывода будет пустой.

Группировка полей формы

Для того, чтобы зрительно отделить одни поля от других используется тег `fieldset`.

Пример:

```
<fieldset>

  <input type="text">

</fieldset>

<fieldset>

  <textarea></textarea>

</fieldset>
```

По умолчанию браузеры отображают результат в виде рамки вокруг этой группы полей, но при помощи CSS можно изменить его внешний вид.

Для каждой группы можно добавить её заголовок. Для этого внутри тега `fieldset` надо поместить тег `legend`:

```
<fieldset>

  <legend>Заголовок группы</legend>

  <input type="text">
```

```
</fieldset>
```

Паттерны значений полей

Для создания поля, принимающего данные определённого формата, используется атрибут `pattern`, в котором с помощью регулярного выражения описывается требуемый формат.

Поле ввода телефона

За ввод телефонных номеров отвечает тип поля `tel`. В такое поле полезно добавлять атрибут `pattern`, чтобы избежать ошибки при вводе данных.

В мобильных браузерах при фокусе на такое поле появляется клавиатура, позволяющая вводить только цифры и символы телефонных номеров.

Подсказка при заполнении полей

Для добавления подсказки к полю формы используется специальный атрибут `placeholder`:

```
<input type="text" placeholder="Текст подсказки">
```

Текст подсказки выводится внутри текстового поля, а при вводе значения — автоматически убирается.

Поля ввода адресов сайтов и email

В HTML5 добавлены два типа полей `email` и `url`, предназначенные для ввода электронной почты и адреса сайта. Эти поля автоматически проверяют формат введённых данных.

Пример записи:

```
<input type="email">
```

```
<input type="url">
```

Поле выбора цвета

В HTML5 добавили новый тип `color`, предназначенный для полей выбора цвета. При клике на такое поле появляется окно с возможностью выбрать цвет из палитры.

Пример записи:

```
<input type="color">
```

Если браузер не поддерживает поле для выбора цвета, то вместо него отображается обычное текстовое поле.

Группировка элементов списка

В теге `select` можно объединять `option` в группы.

Для формирования группы используется тег `optgroup`. Атрибут `label` этого тега определяет заголовок группы.

Пример использования:

```
<select name="variants">

  <optgroup label="Группа вариантов 1">

    <option value="1">Вариант 1</option>

    <option value="2">Вариант 2</option>

    <option value="3">Вариант 3</option>

  </optgroup>

  <optgroup label="Группа вариантов 2">

    <option value="4">Вариант 4</option>

    <option value="5">Вариант 5</option>

    <option value="6">Вариант 6</option>

  </optgroup>

</select>
```

Вложенность групп не ограничена, внутри каждой группы можно вложить другие группы.

Аналогично можно группировать элементы и в списках со множественным выбором.

Запрет редактирования полей

Для того, чтобы сделать поле недоступным для редактирования есть два способа: использование атрибута `readonly` и использование атрибута `disabled`

Атрибут `readonly` не дает пользователю изменять поле (вводить новый текст, модифицировать существующий). Введенное значение можно выделить и скопировать. Данные из этого поля отправляются на сервер.

Атрибут `disabled` не дает пользователю изменять поле (вводить новый текст, модифицировать существующий). Нельзя поставить фокус в это поле, введенное значение нельзя выделять и копировать. Данные из этого поля НЕ отправляются на сервер.

Управление автозаполнением полей

Параметрами автозаполнения можно управлять используя атрибут `autocomplete`. Он может принимать два значения `on` и `off`. Первое включает автозаполнение, второе — отключает.

Пример использования:

```
<input type="text" autocomplete="off">
```

Значение по умолчанию зависит от настроек браузера.

Переключение между полями

При нажатии клавиши `Tab` браузер передает управление (фокус) от одного элемента к другому в том порядке, в котором они были объявлены на странице. Этим порядком можно управлять при помощи атрибута `tabindex`.

Пример записи:

```
<input type="text" tabindex="3">
```

В качестве значения может использоваться любое целое положительное число. Значения выстраиваются последовательно и переход между элементами происходит от меньшего значения к большему. Если представлено отрицательное значение — элемент может быть выделен, однако не участвует в последовательной навигации

Если представлен 0 — элемент может быть выделен и достигнут с помощью последовательной навигации, однако порядок навигации определён платформой

Из соображений доступности не рекомендуется менять порядок навигации по полям ввода по умолчанию.

localStorage

Если данные из формы по каким-то причинам не переданы на сервер, они могут пропасть. Справиться с этой довольно часто встречающейся проблемой поможет `localStorage`.

Суть `localStorage` заключается в том, что в него можно записывать данные, которые будут сохраняться в браузере.

Можно сохранять данные из формы в хранилище при работе с формой, а при загрузке страницы проверять хранилище на наличие данных, и если они есть, то подставлять их в форму. Таким образом можно предотвратить потерю данных при работе с формами.