**2 Работа с формами в HTML5**

**2.1 Задание формы**

Формы в HTML представляют один из способов для ввода и отправки данных. Они являются наиболее популярным способом “обратной связи” с пользователем. С помощью HTML можно создавать как простые формы, предполагающие выбор одного из нескольких ответов, так и сложные формы для заказов или для того, чтобы получить от пользователей страницы какие-либо комментарии и пожелания.

Пользователь может взаимодействовать с веб-сервером разными способами: регистрация и вход на сайтах, ввод личной информации, фильтрация контента, поиск, загрузка файлов. С помощью форм можно предоставить пользователям возможность взаимодействовать с сайтом или приложением, проверять введенную информацию и отправлять данные на сервер [1]. HTML имеет несколько интерактивных элементов управления формами: текстовые поля, переключатели, флажки, выпадающие списки, кнопки отправки. Для реализации этих элементов в HTML используется тег <FORM>.

**<form method=“get|post” action=“url” encctype> Элементы формы и другие элементы HTML</form>**

Элемент <FORM> обозначает документ как форму и определяет границы использования других тегов, размещаемых в форме:

* method: устанавливает метод отправки данных на сервер. Допустимы два значения: post и get;
* значение post позволяет передать данные на веб-сервер через специальные заголовки. А значение get позволяет передать данные через строку запроса;
* action: устанавливает адрес, на который передаются данные формы;
* enctype: устанавливает тип передаваемых данных.

Enctype в свою очередь может принимать следующие значения:

* application/x-www-form-urlencoded: кодировка отправляемых данных по умолчанию;
* multipart/form-data: эта кодировка применяется при отправке файлов;
* text/plain: эта кодировка применяется при отправке простой текстовой информации.

Далее представлен пример создания простейшей формы.

**<body>**

**<form method="post" action="http://localhost:5000/login.js">**

**<input name="regist"/>**

**<input type="submit" value="Войти" />**

**</form>**

**</body>**

Реализация данного кода представлена на рисунке 2.1.

**Изображение выглядит как текст, снимок экрана, Шрифт, число

Контент, сгенерированный ИИ, может содержать ошибки.**

Рисунок 2.1 - Формы в HTML5

В данном примере использовался метод post для отправки данных из формы на сервер. Все значения формы отправляются в теле запроса, а адресом служит строка http://localhost:5000/login.js. Отправка данного запроса приведена на рисунке 2.2.

Изображение выглядит как текст, снимок экрана, Шрифт

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 2.2 - Отправка запроса

Как правило, по указанному адресу работает веб-сервер, который, используя одну из технологий серверной стороны (PHP, NodeJS, ASP.NET и т.д.), он может получать запросы и возвращать ответ.

Часто веб-браузеры запоминают вводимые данные, и при вводе могут выдавать список подсказок из ранее введенных слов.

Изображение выглядит как текст, снимок экрана, дисплей, Шрифт

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 2.3 - Autocomplete в HTML5

С помощью атрибута autocomplete можно отключить авто дополнение:

**<form method="post" autocomplete="off" action="http://localhost:5000/login.js">**

**<input name="regist" /><input name="password" />**

**<input type="submit" value="Войти" /> </form>**

Если необходимо включить автодополнение только для каких-то определенных полей, то можно применить к ним атрибут autocomplete="on":

**<form method="post" autocomplete="off" action="http://localhost:5000/login.js">**

**<input name="regist" />**

**<input name="password" autocomplete="on" />**

**<input type="submit" value="Войти"/>**

**</form>**

Теперь для всей формы, кроме второго поля, будет отключено автодополнение.

**2.2 Элементы форм**

Формы состоят из определенного количества элементов ввода. Все элементы ввода помещаются между тегами <form> и </form>.

Наиболее распространенным элементом ввода является элемент input. Однако реальное действие этого элемента зависит от того, какое значение установлено у его атрибута type. А он может принимать следующие значения:

* text: обычное текстовое поле;
* password: тоже текстовое поле, только вместо вводимых символов отображаются звездочки, поэтому в основном используется для ввода пароля;
* radio: радиокнопка или переключатель. Из группы радиокнопок можно выбрать только одну;
* checkbox: элемент флажок, который может находиться в отмеченном или неотмеченном состоянии;
* hidden: скрытое поле;
* submit: кнопка отправки формы;
* color: поле для ввода цвета;
* date: поле для ввода даты;
* datetime: поле для ввода даты и времени с учетом часового пояса;
* datetime-local: поле для ввода даты и времени без учета часового пояса;
* email: поле для ввода адреса электронной почты;
* month: поле для ввода года и месяца;
* number: поле для ввода чисел;
* range: ползунок для выбора числа из некоторого диапазона;
* tel: поле для ввода телефона;
* time: поле для ввода времени;
* week: поле для ввода года и недели;
* url: поле для ввода адреса url;
* file: поле для выбора отправляемого файла;
* image: создает кнопку в виде картинки.

Кроме элемента input при задании элементов форм можно использовать теги:

* button: кнопка;
* select: выпадающий список;
* label: метка, отображаемая рядом с полем ввода;
* textarea: многострочное текстовое поле.

У всех элементов ввода можно установить атрибуты name и value. Эти атрибуты имеют важное значение.

По атрибуту name можно идентифицировать поле ввода, а атрибут value позволяет установить значение поля ввода. Например:

**<body>**

**<form method="get" action="index.html">**

**<input type="text" name="login" value="Елена"/>**

**<input type="password" name="password"/>**

**<input type="submit" value="Вход" />**

**</form> </body>**

Здесь текстовое поле имеет значение “Елена” (как указано в атрибуте value), поэтому при загрузке веб-страницы в этом поле можно увидеть данный текст (рисунок 2.4).

**Изображение выглядит как текст, снимок экрана, Шрифт, число

Контент, сгенерированный ИИ, может содержать ошибки.**

Рисунок 2.4 - Ввод в форму в HTML5

Поскольку методом отправки данных формы является метод “get”, то данные будут отправляться через строку запроса. Так как в данном случае не важно, как данные будут приниматься, не важен сервер, который получает данные, поэтому в качестве адреса установлена та же самая страница - то есть файл index.html. При отправке можно увидеть введенные данные в строке запроса (рисунок 2.5).

**Изображение выглядит как текст, снимок экрана, Мультимедийное программное обеспечение, Шрифт

Контент, сгенерированный ИИ, может содержать ошибки.**

Рисунок 2.5 - Ввод в форму в HTML5

В строке запроса нас интересует следующий кусочек:

**login=Елена&password=0123456789**

При отправке формы браузер соединяет все данные в набор пар “ключ-значение”. В этом случае две таких пары: login=Елена и password=0123456789. Ключом в этих парах выступает название поля ввода, которое определяется атрибутом name, а значением - собственно то значение, которое введено в поле ввода (или значение атрибута value).

Получив эти данные, сервер легко может узнать, какие значения в какие поля ввода были введены пользователем.

**2.3 Кнопки**

Кнопки в HTML5 представлены элементом button. В зависимости от значения атрибута type можно создать различные типы кнопок:

* submit: кнопка, используемая для отправки формы;
* reset: кнопка сброса значений формы;
* button: кнопка без какого-либо специального назначения.

Например,

**<button type="submit" value="Отправить">**

**< button type="reset" value="Отмена">**

Если кнопка используется для отправки формы, то есть у нее установлен атрибут type=“submit”, то можно задать у нее ряд дополнительных атрибутов:

* form: определяет форму, за которой закреплена кнопка отправки;
* formaction: устанавливает адрес, на который отправляется форма. Если у элемента form задан атрибут action, то он переопределяется;
* formenctype: устанавливает формат отправки данных. Если у элемента form установлен атрибут enctype, то он переопределяется;
* formmethod: устанавливает метод отправки формы (post или get). Если у элемента form установлен атрибут method, то он переопределяется.

Далее представлен пример, использования тега button для создания кнопок отправки и сброса при передаче данных об имени и фамилии пользователя.

**<body><form>**

<h2>Введите свои данные <h2>

**<p><input type="text" name="Imya"/></p>**

**<p><input type="text" name="Famil"/></p>**

**<p> <button type="submit" formmethod="get" formaction="index.html">Отправить</button>**

**<button type="reset">Отмена</button> </p></form></body>**

На рисунке 2.6 представлен пример реализации данного кода.

**Изображение выглядит как текст, снимок экрана, дисплей, программное обеспечение

Контент, сгенерированный ИИ, может содержать ошибки.**

Рисунок 2.6 - Использование тега button для создания кнопок

Кроме элемента button для создания кнопок можно использовать элемент input, у которого атрибут равен submit или reset.

**<input type="submit" value="Отправить">**

**<input type="reset" value="Отмена">**

Если необходимо использовать в качестве кнопки изображение, то можно использовать элемент input с атрибутом type="image".

**<body>**

**<form>**

**<p><input type="text" name="search" />**

**<input type="image" src="search.png" name="submit" /></p>**

**</form>**

**</body>**

Реализация данного кода представлена на рисунке 2.7.

Изображение выглядит как текст, снимок экрана, Шрифт, число

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 2.7 - Кнопка-изображение в HTML5

Кроме наличия изображения в остальном эта кнопка будет аналогична стандартной кнопке отправки input type="submit" или button type="submit".

**2.4 Текстовые поля**

Однострочное текстовое поле создается с помощью элемента input, когда его атрибут type имеет значение text.

**<input type="text" name="login" />**

Текстовое поле используя несколько дополнительных атрибутов:

* dir: направление текста;
* list: список подсказок для ввода в поле;
* maxlength: максимальное количество символов;
* pattern: шаблон, которому должен соответствовать введённый текст;
* placeholder: текст, который будет отображаться по умолчанию в текстовом поле;
* readonly: делает текстовом поле доступным только для чтения;
* required: указывает, что текстовое поле должно иметь значение;
* size: определяет ширину текстового поля в видимых символах;
* value: устанавливает значение по умолчанию в текстовом поле [11].

Далее представлен пример применения некоторых атрибутов при создании текстового поля.

**<body>   <form>**

**<p><input type="text" name="userName" placeholder="ФИО" size="17" /></p>**

**<p> <input type="text" name="userPhone" placeholder="Почта" size="17" maxlength="12" /> </p>**

**<p> <button type="submit">Готово</button>  <button type="reset">Отменить</button> </p> </form> </body>**

Реализация данного кода представлена на рисунке 2.8.

Изображение выглядит как текст, снимок экрана, дисплей, Шрифт

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 2.8 - Текстовые поля в HTML5

В этом примере атрибуты maxlength и size сразу устанавливаются во втором текстовом поле. В данном случае размер - т.е. количество символов, умещающихся в видимом пространстве поля, больше допустимого количества символов. Однако нельзя ввести длину символов, превышающую максимальную.

В этом случае также важно различать атрибуты value и placeholder, хотя оба они устанавливают видимый текст в поле. Placeholder устанавливает подсказку, он неактивен и выделяется серым цветом. Значение value представляет собой текст по умолчанию, введенный в поле.

**<p><input type="text" name="userName" value="Иван" /></p>**

**<p><input type="text" name="userPhone" placeholder="Почта" /></p>**

В данном случае используются два атрибута placeholder обозначения имени и почты (рисунок 2.9).

Изображение выглядит как текст, снимок экрана, электроника, мультимедиа

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 2.9 – Использование placeholder в HTML5

Атрибуты readonly и disabled делают текстовое поле недоступным, однако сопровождаются разным визуальным эффектом. В случае с disabled текстовое поле затеняется (рисунок 2.10).

**<p><input type="text" name="userName" value="Иван" readonly /> </p>**

**<p><input type="text" name="userPhone" value="ivan@mail.ru" disabled /></p>**

Изображение выглядит как текст, снимок экрана, электроника, мультимедиа

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 2.10 - Disabled и readonly в HTML5

Среди атрибутов текстового поля также стоит отметить такой атрибут как list. Атрибут содержит ссылку на элемент datalist, который определяет набор значений, которые появляются в виде подсказок при вводе в текстовое поле. Например:

**<body> <h2>Покупка техники</h2>**

**<form><input list="nameList" type="text" name="model" placeholder="Наименование техники" /></p> <p> <button type="submit">Готово</button> </p> </form>**

**<datalist id="nameList">**

**<option value="Телефон Samsung Galaxy А25" label="20000 руб."/>**

**<option value="Ноутбук** [**Acer Swift Go 16**](https://www.dns-shop.ru/product/0cf33bea715ced20/16-noutbuk-acer-swift-go-16-sfg16-71-70r2-seryj/)**">100000 руб.</option>**

**<option value="Принтер HP375"/>30000 руб.</option>**

**</datalist>**

**</body>**

Пример реализации данного кода представлен на рисунке 2.11.

Изображение выглядит как текст, снимок экрана, программное обеспечение, мультимедиа

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 2.11 – Использование Datalist в HTML5

Атрибут list текстового поля указывает на id элемента datalist. Сам элемент datalist определяет элементы списка, используя вложенные элементы option. И когда пользователь вводит текстовое поле, этот список появляется в виде всплывающей подсказки. Для изменения направления текста в теге input используется атрибут dir, который может принимать два значения: ltr (слева направо) и rtl (справа налево).

**<input type="text" name="username" dir="rtl" />**

**Изображение выглядит как текст, снимок экрана, Шрифт, число

Контент, сгенерированный ИИ, может содержать ошибки.**

Рисунок 2.12 – Направление текста

**2.5 Поле ввода пароля**

Для ввода пароля в HTML используется элемент input с атрибутом type=“password”. Его отличительной чертой является то, что вводимые символы маскируются точками.

**<h2> Введите пароль </h2>**

**<form><p> <input type="text" name="login" /></p>  <p>**

**<input type="password" name="password" placeholder="пароль"/>**

**<input type="submit" value="Авторизация" /></form>**

Изображение выглядит как текст, снимок экрана, программное обеспечение, мультимедиа

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 2.13 - Ввод пароля в HTML5

По своим атрибутам поле типа password не отличается от поля типа text. При использовании этого поля следует помнить, что пароль или любая другая информация, которая вводится в поле типа password, будет передаваться в виде ASCII-символа и будет доступна для просмотра.

**2.6 Метки и автофокус**

Вместе с полями ввода нередко используются метки, которые представлены элементом label. Метки создают аннотацию или заголовок к полю ввода, указывают, для чего это поле предназначено.Для связи с полем ввода метка имеет атрибут for, который указывает на id поля ввода.

**<body> <h1>Укажите цвет и форму дивана</h1> <form> <p>**

**<label for="color">Цвет: </label>**

**<input type="text" id="color" name="color" /> </p> <p>**

**<label for="form">Форма: </label>**

**<input type="name" id="form" name="form" /> </p> <p>**

**<button type="submit">Отправить</button> </p>**

**</form> </body>**

Так, текстовое поле здесь имеет атрибут id=" firstname ". Поэтому у связанной с ним метки устанавливается атрибут for=" firstname ". Нажатие на эту метку позволяет перевести фокус на текстовое поле для ввода фамилии (рисунок 2.14).

Изображение выглядит как текст, снимок экрана, программное обеспечение, мультимедиа

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 2.14 – Использование Label в HTML5

Собственно, на этом роль меток и заканчивается. Также можно установить автофокус по умолчанию на какое-либо поле ввода. Для этого применяется атрибут autofocus.

**<h1>Укажите цвет и форму дивана</h1>**

**<form>**

**<p> <label for="color">Цвет: </label>**

**<input type="text" autofocus id="color" name="color" /> </p>**

**<p> <label for="form">Форма: </label>**

**<input type="name" id="form" name="form" /> </p>**

**<p> <button type="submit">Отправить</button> </p>**

**</form>**

Здесь при запуске страницы фокус сразу же переходит на текстовое поле (рисунок 2.15).

Изображение выглядит как текст, снимок экрана, программное обеспечение, мультимедиа

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 2.15 – Использование автофокусов в HTML

**2.7 Элементы для ввода чисел**

**2.7.1 Простое числовое поле**

Для ввода чисел в HTML5 используется элемент input с атрибутом type=“number”. Он создает числовое поле, которое можно настроить с помощью следующих атрибутов:

* min: минимально допустимое значение;
* max: максимально допустимое значение;
* readonly: доступно только для чтения;
* required: указывает, что данное поле обязательно должно иметь значение;
* step: значение, на которое будет увеличиваться число в поле;
* value: значение по умолчанию.

Далее представлен пример работы с числовыми полями.

**<body> <h1>Укажите ваш возраст</h1>**

**<form><p> <label for="age">Возраст: </label>**

**<input type="number" step="1" min="1" max="100" value="10" id="age" name="age"/>  </p>**

**<p>   <button type="submit">Отправить</button> </p> </form>    </body>**

Реализация данного кода представлена на рисунке 2.16.

Изображение выглядит как текст, снимок экрана, мультимедиа, программное обеспечение

AI-generated content may be incorrect.

Рисунок 2.16 - Ввод в числовое поле

Здесь числовое поле по умолчанию имеет значение 10 (value=“10”), минимально допустимое значение, которое можно ввести, - 1, а максимальное допустимое значение - 100. В зависимости от браузера визуализация этого поля может отличаться.

**<body><h1>Сколько хотите купить мороженного?</h1>**

**<form> <p> <label for="num">Количество: </label>**

**<input type="number" list="numList"**

**step="10" min="30" max="100" value="100" id="num" name="num"/> </p>**

**<p> <button type="submit">Готово</button>  </p>**

**</form>   <datalist id="numList">**

**<option value="1" />  <option value="2" />**

**<option value="3" /> <option value="4" />**

**<option value="5" />   </datalist>   </body>**

Пример реализации данного кода представлен на рисунке 2.17.

Изображение выглядит как текст, снимок экрана, программное обеспечение, мультимедиа

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 2.17 - Ввод чисел из datalist в HTML5

**2.7.2 Ползунок**

Ползунок представляет шкалу, на которой можно выбрать одно из значений. Для создания ползунка применяется элемент input с атрибутом type="range". Во многом ползунок похож на простое поле для ввода чисел. Он также имеет атрибуты min, max, step и value.

**<body>  <form>**

**<p> <label for="num">Количество:</label>**

**1<input type="range" step="1" min="0" max="100" value="10" id="num" name="num"/>100 </p>**

**<p> <button type="submit">Готово</button> </p> </form> </body>**

Реализация данного кода представлен на рисунке 2.18.

Изображение выглядит как текст, снимок экрана, программное обеспечение, мультимедиа

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 2.18 - Ползунок в HTML5 в Яндекс браузере

Аналогично, как и атрибут "number" тега input, атрибут "range" по-разному отображается в разных браузерах (рисунок 2.19).

Изображение выглядит как текст, программное обеспечение, снимок экрана

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 2.19 - Ползунок в HTML5 в Edge

**2.8 Флажки и переключатели**

**2.8.1 Флажок**

Флажок представляет элемент, который может находиться в двух состояниях: отмеченном и неотмеченном. Флажок создается с помощью элемента input с атрибутом type=“checkbox”.

**<body> <h2> Выберите любимого поэта</h2>**

**<form> <p> <input type="checkbox" checked name="pushkin"/>Пушкин</p>**

**<p> <input type="checkbox" value="lermontov" />**

**<label for="lermontov">Лермонтов</label> </p>**

**<p> <input type="checkbox" name="blok"/>Блок </p>**

**<p> <button type="submit">Готово</button> </p>**

**</form> </body>**

Реализация данного кода представлена на рисунке 2.20.

Изображение выглядит как текст, снимок экрана, программное обеспечение, мультимедиа

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 2.20 – Флажок (чек-бокс) в HTML5

Атрибут checked позволяет установить флажок в отмеченное состояние при загрузке страницы.

**<input type="checkbox" value="lermontov" / checked>**

**2.8.2 Переключатели**

Переключатели или радиокнопки похожи на флажки, они также могут находиться в отмеченном или неотмеченном состоянии. Только для переключателей можно создать одну группу, в которой одновременно можно выбрать только один переключатель, так как в отличии от переключателя используется одно имя для присвоения значения.

**<body><form> <h2>Укажите цвет</h2>**

**<p> <input type="radio" value="white" checked name="color"/>белый </p> <p>**

**<input type="radio" value="red" name="color"/>красный</p><h2>Выберите цветы</h2><p>**

**<input type="radio" value="rose" checked name="flower"/>Розы </p> <p>**

**<input type="radio" value="tulp" name="flower"/>Тюльпаны</p> <p>**

**<button type="submit">Готово</button> </p> </form> </body>**

Реализация данного кода представлена на рисунке 2.21.

Изображение выглядит как текст, снимок экрана, программное обеспечение, мультимедиа

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 2.21 - Радиокнопки в HTML5

Для создания радиокнопки надо указать атрибут type=“radio”. И теперь другой атрибут name указывает не на имя элемента, а на имя группы, к которой принадлежит элемент-радиокнопка. В данном случае две группы радиокнопок: color и flower. Из каждой группы можно выбрать только один переключатель. Опять же чтобы отметить радиокнопку, у нее устанавливается атрибут checked.

Важное значение играет атрибут value, который при отправке формы позволяет серверу определить, какой именно переключатель был отмечен.

**2.9 Элементы для ввода цвета, url, email, телефона**

**2.9.1 Установка цвета**

За установку цвета в HTML5 отвечает специальный элемент input с типом color.

**<label for="tsvet">Выберите цвет</label>**

**<input type="color" id="tsvet" name="favcolor" />**

Элемент отображает выбранный цвет. А при нажатии на него появляется специальное диалоговое окно для установки цвета [13].

Изображение выглядит как текст, снимок экрана, мультимедиа, программное обеспечение

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 2.22 - Элемент установки цвета в HTML5

Значением этого элемента является числовой шестнадцатеричный код выбранного цвета. С помощью элемента datalist можно указать набор цветов, из которых пользователь сможет выбрать нужный.

**<label for="tsvet">Выбор цвета</label>**

**<input type="color" list="colors" id="tsvet" name="tsvet" />**

**<datalist id="colors">**

**<option value="#FF1493" label="DeepPink">**

**<option value="#20B2AA" label="LightSeaGreen">**

**<option value="#1E90FF" label="DodgerBlue">**

**</datalist>**

Реализация данного кода представлена на рисунке 2.23.

Изображение выглядит как снимок экрана, текст, программное обеспечение, мультимедиа

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 2.23 - Выбор цвета в HTML5

Каждый элемент option в datalist должен в качестве значения принимать шестнадцатеричный код цвета, например, "#0000FF". После выбора цвета данный числовой код устанавливается в качестве значения в элементе input.

**2.9.2 Поля для ввода url, email, телефона**

Различные поля ввода input предназначены для ввода таких данных, как URL-адрес, адрес электронной почты и номер телефона. Они однотипны и во многом отличаются только тем, что в качестве атрибута type принимают значения email, tel и url соответственно. Для их настройки можно использовать те же атрибуты, что и для обычного текстового поля:

* maxlength: максимальное количество символов, разрешенное в поле;
* pattern: определяет шаблон, которому должен соответствовать введенный текст;
* placeholder: определяет текст, который будет отображаться в поле по умолчанию;
* readonly: делает текстовое поле доступным только для чтения;
* required: указывает, что текстовое поле должно иметь значение;
* size: определяет ширину поля в видимых символах;
* value: определяет значение поля по умолчанию;
* list: определяет привязку к элементу datalist со списком возможных значений.

**<body>**

**<form><p> <label for="email">Почта: </label><input type="email" placeholder="person@mail.ru" id="email" name="email"/> </p>**

**<p> <label for="url">URL: </label><input type="url" id="url" name="url"/> </p>**

**<p> <label for="phone">Телефон: </label>**

**<input type="tel" placeholder="+7 (XXX) - XXX - XX - XX" id="phone" name="phone"/>**

**</p> <p> <button type="submit">Готово</button>**

**</p> </form> </body>**

Реализация данного кода представлена на рисунке 2.24.

**Изображение выглядит как текст, снимок экрана, Шрифт, дисплей

Контент, сгенерированный ИИ, может содержать ошибки.**

Рисунок 2.24 - Ввод почты, url, телефона в HTML5

Основное преимущество таких полей ввода перед обычными текстовыми полями заключается в том, что поля ввода электронной почты, URL-адреса и телефона используют соответствующий шаблон для проверки ввода. Например, если вводится неверное значение в какое-либо поле и отправляется форма, браузер может отобразить сообщение о неправильном вводе, а форма не будет отправлена (рисунок 2.25).

Изображение выглядит как текст, снимок экрана, программное обеспечение, Шрифт

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 2.25 - Валидация почты в HTML5

**2.10 Элементы для ввода даты и времени**

Для работы с датами и временем в HTML5 предназначено несколько типов элементов input:

* datetime-local: устанавливает дату и время;
* date: устанавливает дату; month: устанавливает текущий месяц и год;
* time: устанавливает время; week: устанавливает текущую неделю.

Далее представлен пример установки даты.

**<body><form> <p><label for="name"> ФИО: </label><input type="text" id="name" name="name"/> </p>**

**<p><label for="date">Дата рождения: </label> <input type="date" id="date" name="date"/></p> <p> <button type="submit">Готово</button> </p> </form> </body>**

Реализация данного кода представлена на рисунке 2.26.

Изображение выглядит как текст, снимок экрана, программное обеспечение, мультимедиа

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 2.26 - Установка даты в HTML5

В данном случае прим вводе в поле для даты будет открываться календарь. Аналогично многим расширениям HTML5 работа с датой по-разному отображается в разных браузерах.

Далее приведены примеры использования остальных элементов.

**<body>   <form>**

**<p> <label for="week">Неделя: </label><input type="week" name="week" id="week" />  </p>**

**<p> <label for="localdate">Дата и время: </label> <input type="datetime-local" id="localdate" name="date"/></p><p>**

**<label for="month">Месяц: </label> <input type="month" id="month" name="month"/></p>**

**<p> <label for="time">Время: </label><input type="time" id="time" name="time"/></p>**

**<p><button type="submit">Готово</button></p> </form></body>**

Реализация данного кода представлена на рисунке 2.27.

Изображение выглядит как текст, снимок экрана, программное обеспечение, мультимедиа

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 2.27 – Установка даты и времени в HTML5

При использовании этих элементов также надо учитывать, что Firefox поддерживает только элементы date и time, для остальных создаются обычные текстовые поля. А IE11 и вовсе не поддерживают эти элементы.

**2.11 Отправка файлов**

За выбор файлов на форме в HTML5 отвечает элемент input с атрибутом type="file".

**<body>**

**<form enctype="multipart/form-data" method="post" action="**[**http://localhost:5000/postfile.js**](http://localhost:8080/postfile.php)**">**

**<p> <input type="file" name="file" /></p>**

**<p><input type="submit" value="Готово" />**

**</p></form> </body>**

Реализация данного кода представлена на рисунке 2.28.

Изображение выглядит как текст, снимок экрана, Шрифт, число

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 2.28 - Выбор файла в HTML5

При нажатии на кнопку “Выберите файл” открывается диалоговое окно для выбора файла. А после выбора рядом с кнопкой отображается имя выбранного файла. Для отправки файла на сервер форма должна иметь атрибут enctype="multipart/form-data". С помощью атрибутов можно настроить элементы выбора файла:

* accept: устанавливает тип файл, которые допустимы для выбора;
* multiple: позволяет выбирать множество файлов;
* required: требует обязательной установки файла.

Далее приведен пример множественного выбора файлов.

**<form enctype="multipart/form-data" method="post" action="http://localhost:5000/postfile.js">**

**<p> <input type="file" name="file" multiple /> </p>**

**<p><input type="submit" value="Готово" /> </p>**

**</form>**

Реализация данного кода представлена на рисунке 2.29.

Изображение выглядит как текст, снимок экрана, Шрифт, число

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 2.29 - Множественный выбор файлов в HTML5

При нажатии на кнопку открывается диалоговое окно для выбора файлов, зажав клавишу CTRL или Shift, можно выбрать несколько файлов, а после выбора рядом с кнопкой отобразится количество выбранных файлов:

**2.12 Список select**

Элемент select создает список. В зависимости от настроек это может быть выпадающий список для выбора одного элемента, либо раскрытый список, в котором можно выбрать сразу несколько элементов. Далее представлен пример создания выпадающего списка.

**<body> <form method="get"> <p>**

**<label for="name">Наименование</label>**

**<select id="name" name="name">**

**<option value="pen">Ручка</option>**

**<option value="pencil">Карандаш</option>**

**<option value="eraser">Ластик</option>**

**<option value="liner">Линейка</option> </select>**

**</p> <p> <input type="submit" value="Готово" /> </p>**

**</form> </body>**

Реализация данного кода представлена на рисунке 2.30.

Изображение выглядит как текст, снимок экрана, программное обеспечение, мультимедиа

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 2.30 - Выпадающий список в HTML5

Внутри элемента select помещаются элементы option - элементы списка. Каждый элемент option содержит атрибут value, который хранит значение элемента. При этом значение элемента option не обязательно должно совпадать с отображаемым им текстом. Например:

**<option value="pen">Ручка</option>**

С помощью атрибута selected можно установить выбранный по умолчанию элемент - это необязательно должен быть первый элемент в списке:

**<select id="name" name="name">**

**<option value="pen">Ручка</option>**

**<option value="pencil">Карандаш</option>**

**<option value="eraser">Ластик</option>**

**<option value="liner">Линейка</option> </select>**

**</select>**

Для создания списка с множественным выбором к элементу select необходимо добавить атрибут multiple.

**<body> <form method="get">  <p>**

**<label for="name">Наименование техники:</label> <br/>**

**<select multiple id="name" name="name">**

**<option value="phone">Телефон</option>**

**<option value="laptop">Ноутбук</option>**

**<option value="plan">Планшет</option>**

**<option value="laptop">Принтер</option>**

**<option value="mouse">Мышь</option>**

**</select> </p>  <p> <input type="submit" value="Готово" /> </p>  </form></body>**

Зажав клавишу Ctrl, можно выбрать в таком списке несколько элементов:

Реализация данного кода представлена на рисунке 2.31.

Изображение выглядит как текст, снимок экрана, программное обеспечение, мультимедиа

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 2.31 - Multiple select в HTML5

Select также позволяет группировать элементы с помощью тега <optgroup>. Далее представлен пример использования группировки.

**<body> <form method="get"> <p>**

**<label for="name">Наименование</label> <select id="name" name="name">**

**<optgroup label="Канцтовары">**

**<option value="pen">Ручка</option>**

**<option value="pencil">Карандаш</option>**

**<option value="eraser">Ластик</option>**

**<option value="liner">Линейка</option> </select>**

**</optgroup><optgroup label="Учебники">**

**<option value="book">Учебник по математике</option> ">**

**<option value="book">Учебник по физике</option> </optgroup> </select> </p>**

**<p> <input type="submit" value="Готово" /></p> </form> </body**

Реализация данного кода представлена на рисунке 2.32.

Изображение выглядит как текст, снимок экрана, программное обеспечение, мультимедиа

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 2.32 - Optgroup в HTML5

Использование групп элементов применимо как к выпадающему списку, так и к списку со множественным выбором.

**2.13 Textarea**

Элемент <input type="text"/> позволяет создавать простое однострочное текстовое поле. Однако возможностей этого элемента по вводу текста бывает недостаточно, и в этой ситуации можно использовать многострочное текстовое поле, представленное элементом textarea. С помощью дополнительных атрибутов cols и rows можно задать соответственно количество столбцов и строк.

Далее представлен пример использования тег <textarea>.

**<body> <form method="get">**

**<p> <label for="comment">Контент</label><br/>**

**<textarea id="comment" name="comment" placeholder="Текст поста"**

**cols="30" rows="7"> </textarea></p>**

**<p> <input type="submit" value="Опубликовать" /></p>**

**</form>   </body>**

Реализация данного кода представлена на рисунке 2.33.

Изображение выглядит как текст, снимок экрана, дисплей, Шрифт

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 2.33 – Настройка размера блока textarea в HTML5

**2.14 Валидация форм**

Для создания форм в распоряжении имеются различные элементы, которые можно использовать. Можно вводить в них различные значения. Однако нередко пользователи вводят не совсем корректные значения: например, ожидается ввод чисел, а пользователь вводит буквы и т.д. И для предупреждения и проверки некорректного ввода в HTML5 существует механизм валидации.

Преимущество использования валидации в HTML5 заключается в том, что пользователь после некорректного ввода может сразу получить сообщение об ошибке и внести соответствующие изменения в введенные данные.

Для создания валидации у элементов форм HTML5 используются атрибуты:

* required: требует обязательного ввода значения. Для элементов textarea, select, input (с типом text, password, checkbox, radio, file, datetime-local, date, month, time, week, number, email, url, search, tel);
* min и max: минимально и максимально допустимые значения. Для элемента input с типом datetime-local, date, month, time, week, number, range;
* pattern: задает шаблон, которому должны соответствовать вводимые данные. Для элемента input с типом text, password, email, url, search, tel.

**2.14.1 Атрибут required**

Атрибут required требует обязательного наличия значения.

**<body><form method="get">  <p> <label for="login">Почта:</label>**

**<input type="text" required id="login" name="login" />**

**</p> <p>   <label for="password">Пароль:</label>**

**<input type="password" required id="password" name="password" />**

**</p>  <p>  <input type="submit" value="Готово" />  </p>**

**</form>  </body>**

Реализация данного кода представлена на рисунке 2.34.

Изображение выглядит как текст, снимок экрана, дисплей, Шрифт

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 2.34 - Атрибут required в HTML5

Если не ввести в эти поля никаких данных, оставив их пустыми, и нажать на кнопку отправки, то браузер высветит сообщения об ошибке, а данные не будут отправлены на сервер. В зависимости от браузера визуализация сообщения может несколько отличаться. Также границы некорректного поля ввода могут окрашиваться в красный цвет. Например, поведение при отправке пустых сообщений в Microsoft Edge.

**2.14.2 Атрибуты max и min**

Для ограничения диапазона вводимых значений применяются атрибуты max и min.

**<body> <form method="get">**

**<p> <label for="age">Возраст:</label>**

**<input type="number" min="1" max="100" value="18" id="age" name="age"/> </p>**

**<p> <input type="submit" value="Готово" />  </p>   </form>  </body>**

Реализация данного кода представлена на рисунке 2.35.

Изображение выглядит как текст, снимок экрана, Шрифт, мультимедиа

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 2.35 - Ограничение диапазона чисел в HTML5

**2.14.3 Синтаксис регулярных выражений**

**2.14.3.1 Метасимволы**

Регулярные выражения могут использовать метасимволы - символы, которые имеют определенный смысл:

* \d: соответствует любой цифре от 0 до 9;
* \D: соответствует любому символу, который не является цифрой;
* \w: соответствует любой букве, цифре или символу подчеркивания (диапазоны A-Z, a-z, 0-9);
* \W: соответствует любому символу, который не является буквой, цифрой или символом подчеркивания (то есть не находится в следующих диапазонах A-Z, a-z, 0-9);
* \s: соответствует пробелу;
* \S: соответствует любому символу, который не является пробелом;
* .: соответствует любому символу.

Здесь надо заметить, что метасимвол \w применяется только для букв латинского алфавита, кириллические символы для него не подходят.

Так, стандартный формат номера телефона +1-234-567-8901 соответствует регулярному выражению \d-\d\d\d-\d\d\d-\d\d\d\d. Например, числа номера заменяются нулями:

**let phoneNumber = "+1-234-567-8901";**

**let myExp = /\d-\d\d\d-\d\d\d-\d\d\d\d/;**

**phoneNumber = phoneNumber.replace(myExp, "00000000000");**

**document.write(phoneNumber);**

**2.14.3.2 Модификаторы**

Кроме выше рассмотренных элементов регулярных выражений есть еще одна группа комбинаций, которая указывает, как символы в строке будут повторяться. Такие комбинации еще называют модификаторами:

* {n}: соответствует n-ому количеству повторений предыдущего символа. Например, h{3} соответствует подстроке "hhh";
* {n,}: соответствует n и более количеству повторений предыдущего символа. Например, h{3,} соответствует подстрокам "hhh", "hhhh", "hhhhh" и т.д.;
* {n,m}: соответствует от n до m повторений предыдущего символа. Например, h {2, 4} соответствует подстрокам "hh", "hhh", "hhhh";
* ?: соответствует одному вхождению предыдущего символа в подстроку или его отсутствию в подстроке. Например, /h?ome/ соответствует подстрокам "home" и "ome";
* +: соответствует одному и более повторений предыдущего символа;
* \*: соответствует любому количеству повторений или отсутствию предыдущего символа;

Например, берется номер тот же телефона. Ему соответствует регулярное выражение \d-\d\d\d-\d\d\d-\d\d\d\d. Однако с помощью выше рассмотренных комбинаций можно его упростить: \d-\d{3}-\d{3}-\d{4}

Также надо отметить, что так как символы ?, +, \* имеют особый смысл в регулярных выражениях, то чтобы их использовать в обычным для них значении (например, надо заменить знак плюс в строке на минус), то данные символы надо экранировать с помощью слеша:

**let phoneNumber = "+1-234-567-8901";**

**let myExp = /\+\d-\d{3}-\d{3}-\d{4}/;**

**phoneNumber = phoneNumber.replace(myExp, "80000000000");**

**console.log(phoneNumber);**

**2.14.4 Атрибут pattern**

Атрибут pattern задает шаблон, которому должны соответствовать данные. Для определения шаблона используется язык регулярных выражений.

**<body> <form method="get">**

**<p> <h2>Укажите телефон</h2>**

**<label for="phone">Телефон:</label>**

**<input type="text" placeholder="+7999-999-99-99"**

**pattern="\+\d {4}-\d {3}-\d {2}-\d {2}" id="phone" name="phone" /> </p>**

**<p> <input type="submit" value="Готово" />   </p> </form>  </body>**

Здесь для ввода номера телефона используется регулярное выражение \+\d {4}-\d {3}-\d {2}-\d {2}. Оно означает, что первым элементом в номере должен идти знак плюс +. Выражение \d представляет любую цифру от 0 до 9. Выражение \d {3} означает три подряд идущих цифры, а \d {4} - четыре цифры подряд. То есть это выражение будет соответствовать номеру телефона в формате "+7999-000-00-00".

Если ввести данные, которые не соответствуют этому шаблону, и нажать на отправку, то браузер отобразит ошибку (рисунок 2.36).

Изображение выглядит как текст, снимок экрана, программное обеспечение, мультимедиа

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 2.36 - Регулярные выражения в HTML5

**2.14.5 Отключение валидации**

Не всегда валидация является желаемой, иногда требуется ее отключить. И в этом случае можно использовать либо у элемента формы атрибут novalidate, либо у кнопки отправки атрибут formnovalidate.

**<head> <body> <form novalidate method="get">**

**<p>  <label for="phone">Телефон:</label>**

**<input type="text" placeholder="+1-234-567-8901"**

**pattern="\+\d-\d{3}-\d{3}-\d{4}" id="phone" name="phone" />  </p>**

**<p> <input type="submit" value="Готово" formnovalidate /> </p> </form> </body>**

**2.14.6 Элементы fieldset и legend**

Для группировки элементов формы нередко применяется элемент fieldset. Он создает границу вокруг вложенных элементов, как бы создавая из них группу. Вместе с ним используется элемент legend, который устанавливает заголовок для группы элементов [14].

**<body>  <h2>Вход на сайт</h2>**

**<form> <fieldset>  <legend>Введите данные:</legend>**

**<label for="login">Логин:</label><br>**

**<input type="text" name="login" id="login" /><br>**

**<label for="password">Пароль:</label><br>**

**<input type="password" name="password" id="password" /><br>**

**<input type="submit" value="Авторизация"> </fieldset></form> </body>**

Реализация данного кода представлена на рисунке 2.37.

Изображение выглядит как текст, снимок экрана, программное обеспечение

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 2.37 - Группировка элементов формы в HTML5

При необходимости можно создать на одной форме несколько групп с помощью элементов fieldset.

При создании форм на сайтах используется сочетание рассмотренных элементов управления формами. Далее представлен пример использования всех элементов при создании анкеты студента.

**<form> <legend>Форма</legend>  
 <label for="login">\*Фамилия:</label><br />  
 <input type="text" name="name" id="name" required /><br />  
 <label for="login">\*Имя:</label><br />  
 <input type="text" name="surname" id="surname" required /><br />  
 <label for="login">Отчество:</label><br />  
 <input type="text" name="patronymic" id="patronymic" /><br /> <hr/>  
 <label for="sex">\*Пол:</label><br />  
 <input type="radio" value="man" checked name="pol" />мужской  
 <input type="radio" value="woman" name="pol" />женский <hr/>  
 <p> <label for="date">\*Дата рождения: </label>  
 <input type="date" id="date" name="date" required /> </p><hr />  
 <p> <label for="email">\*Почта: </label>  
 <input type="email" placeholder="person@mail.ru" id="email" name="email"  
 required /> </p><hr />  
 <label for="phone">\*Телефон:</label>  
 <input type="text" pattern="\+7[0-9]{10}" placeholder="+7\_\_\_\_\_\_\_\_\_\_"  
 id="phone" name="phone" required/><br /><hr />  
 <p><label for="name">\*Ваш курс:</label>  
 <select id="name" name="name" required>  
 <option value="1 курс">Первый</option>  
 <option value="2 курс">Второй</option>  
 <option value="3 курс">Третий</option>  
 <option value="4 курс">Четвертый</option> </select> </p><hr />  
 <label>Выберете направления подготовки:</label><br />  
 <input type="checkbox" name="blok" />ПиНЖ<br />  
 <input type="checkbox" name="blok" />ИВТ<br />  
 <input type="checkbox" name="blok" />САПР<br />  
 <p> <label for="mastery">Уровень освоения программирования:</label>  
 1<input type="range" step="1" min="0" max="10" value="5" id="mastery" name="mastery"/>10 </p>  
 <p> <label for="grade">Средняя оценка по предметам</label>  
 <input type="number" step="0.1" min="1" max="5" value="5" id="grade" name="grade"/> </p>  
 <label for="comment">О себе:</label><br />  
 <textarea id="comment" name="comment" placeholder="Расскажите о себе..."  
 cols="30" rows="3" ></textarea> <hr />  
 <label for="login">\*Логин:</label><br />  
 <input type="text" name="login" id="login" required /><br /><hr />  
 <label for="password">\*Пароль:</label><br />  
 <input type="password" name="password" id="password" required /><br /> <hr />  
 <input type="submit" value="Отправить" /> </fieldset> </form>**

Реализация данного кода представлена на рисунке 2.38.

Изображение выглядит как текст, снимок экрана, программное обеспечение, число

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 2.38 – Пример использования всех элементов управления формами