# Построение форм

Формы являются неотъемлемой частью Интернета, так как они предлагают сайтам метод сбора информации от пользователей и обработки запросов, а также элементы управления практически для любого мыслимого применения. С помощью элементов управления или полей, формы могут запросить небольшой объём информации — часто это поисковый запрос, имя пользователя или пароль. Или большой объём информации — возможно, данные о посылке, платёжная информация или предложение работы.

## Инициализация формы

Чтобы добавить форму на страницу мы будем использовать элемент <form>. Данный элемент определяет, где на странице появятся элементы управления. Кроме того, элемент <form> обёртывает все элементы включенные в форму, подобно элементу <div>.

<form action="/login" method="post">

...

</form>

К элементу <form> может применяться горстка различных атрибутов, наиболее распространёнными из которых являются action и method. Атрибут action содержит URL, на который информация в форме будет отправлена для обработки сервером. Атрибут method является методом HTTP, который должны использовать браузеры для отправки данных формы. Оба эти атрибута <form> имеют отношение к отправке и обработке данных.

### Текстовые поля

Одним из основных элементов, используемых для получения текста от пользователей, является элемент <input>. Данный элемент включает атрибут type для определения, какой тип информации будет получен в элементе управления. Наиболее популярное значение атрибута type — это text, который обозначает ввод одной строки текста.

Наряду с установкой атрибута type, хорошей практикой будет также дать элементу <input> атрибут name. Значение атрибута name применяется в качестве имени элемента управления и отправляется вместе с входными данными на сервер.

<input type="text" name="username">

Первоначально было только два текстовых значения атрибута type — text и password (для ввода паролей), однако HTML5 привёз с собой несколько новых значений атрибута type.

Эти значения были добавлены, чтобы обеспечить чёткое смысловое значение для полей ввода, а также предоставить лучшее управление пользователям. Если браузер не понимает одно из этих HTML5-значений атрибута type, он автоматически вернётся к значению text. Ниже приведён список новых типов HTML5.

* color
* date
* month
* number
* range
* search
* week

<input type="date" name="birthday">

### <textarea>

Ещё одним элементом, используемым для сбора текстовых данных, является элемент <textarea>. Он отличается от элемента <input> тем, что может принимать большие отрывки текста в несколько строк. Элемент <textarea> также содержит начальный и конечный теги, которые могут обернуть простой текст. Поскольку <textarea> принимает только один тип значения, атрибут type здесь не применим, но атрибут name по прежнему используется.

<textarea name="comment">Добавьте сюда свой комментарий</textarea>

### Переключатели

Переключатели — это простой способ, позволяющий пользователям сделать быстрый выбор из небольшого списка вариантов. Переключатели дают пользователю выбрать только один вариант в отличие от нескольких.

Чтобы создать переключатель, используется элемент <input> со значением radio у атрибута type. Каждый переключатель должен иметь одинаковое значение атрибута name, чтобы все они в группе были связаны друг с другом.

С текстовыми полями их значение определяется тем, что пользователь в них набирает; с переключателями пользователь делает множественный выбор. Таким образом, мы должны определить входное значение. Используя атрибут value мы можем установить конкретное значение для каждого элемента <input>.

Кроме того, для предварительного выбора переключателя мы можем использовать логический атрибут checked.

<input type="radio" name="day" value="Friday" checked> Пятница

<input type="radio" name="day" value="Saturday"> Суббота

<input type="radio" name="day" value="Sunday"> Воскресенье

### Флажки

Флажки очень похожи на переключатели. Они используют те же атрибуты и шаблоны, за исключением значения атрибута type. Разница между ними состоит в том, что флажки позволяют пользователям выбрать несколько значений и связать их все с одним именем, в то время как переключатели ограничивают пользователей одним значением.

<input type="checkbox" name="day" value="Friday" checked> Пятница

<input type="checkbox" name="day" value="Saturday"> Суббота

<input type="checkbox" name="day" value="Sunday"> Воскресенье

### Выпадающие списки

Выпадающие списки являются идеальным способом, чтобы практическим образом предоставить пользователям длинный список вариантов. Длинный столбец переключателей рядом со списком разных вариантов не только визуально непривлекателен, но кроме того сложен и труден для понимания, особенно на мобильном устройстве. Выпадающие списки, с другой стороны, обеспечивают идеальный формат для длинного списка вариантов.

Для создания выпадающего списка мы будем применять элементы <select> и <option>. Элемент <select> оборачивает все пункты меню, а каждый пункт меню размечен с помощью элемента <option>.

Атрибут name располагается в элементе <select>, а атрибут value располагается в элементах <option>, вложенных в элемент <select>. Таким образом, атрибут value в каждом элементе <option> связан с атрибутом name элемента <select>.

Каждый элемент <option> оборачивает текст (который виден пользователям) отдельного пункта в списке.

Подобно логическому атрибуту checked у переключателей и флажков, для выпадающего меню можно использовать логический атрибут selected, чтобы предварительно выделить пункт для пользователей.

<select name="day"> multiple

<option value="Friday" selected>Пятница</option>

<option value="Saturday">Суббота</option>

<option value="Sunday">Воскресенье</option>

</select>

### Кнопка для отправки

Кнопка для отправки в виде элемента <input> является самодостаточной и не может оборачивать любой другой контент. Если хочется иметь больше контроля над структурой и дизайном поля, наряду с возможностью обернуть другие элементы — тогда может быть использован элемент <button>.

Элемент <button> выполняет то же самое, что и элемент <input> со значением submit у атрибута type. Однако, он включает в себя открывающий и закрывающий теги, которые могут обернуть другие элементы. По умолчанию, элемент <button> действует, как если у атрибута type задано значение submit, поэтому атрибут type и его значение можно по желанию опустить.

Вместо использования атрибута value для управления текстом в кнопке для отправки, будет показан текст, который пишется между открывающим и закрывающим тегами элемента <button>.

<button name="submit">

<strong>Отправьте нам</strong> сообщение

</button>

### Поле для файла

Чтобы позволить пользователям добавить файл в форму, вроде прикрепления файла к письму, используйте значение file атрибута type.

<input type="file" name="file">

## Организация элементов формы

С помощью <label>, <fieldset> и <legend> мы можем лучше организовать формы и направлять пользователей правильно их завершать.

### <label>

Элемент <label> содержит подписи или заголовки для управления формой, однозначно связывая их вместе, создавая тем самым доступную форму для всех пользователей и вспомогательных технологий. <label> должны включать в себя текст, описывающий поля к которым они относятся.

<label> могут включать в себя атрибут for, его значение должно быть таким же, как значение атрибута id у элемента, с которым связан <label>. Соответствие значений атрибутов for и id связывает два элемента вместе, что позволяет пользователям нажать на <label> и передать фокус нужному полю формы.

<label for="username">Имя пользователя</label>

<input type="text" name="username" id="username">

### <fieldset>

<fieldset> группирует поля формы в организованные разделы. Подобно <section> или иным структурным элементам, но <fieldset> является блочным элементом, который оборачивает связанные элементы, в частности, в <form>, для их лучшей организации. <fieldset> по умолчанию также включают в себя границы контура, которые могут быть изменены с помощью CSS.

<fieldset>

<label>

Имя пользователя

<input type="text" name="username">

</label>

<label>

Пароль

<input type="text" name="password">

</label>

</fieldset>

### <legend>

Элемент <legend> предоставляет подпись или заголовок для элемента <fieldset>. Элемент <legend> оборачивает текст, описывающий элементы управления формы, которые находятся внутри <fieldset>. Разметка должна включать в себя элемент <legend> сразу после открывающего тега <fieldset>. На странице подпись появится в левом верхнем углу рамки <fieldset>.

<fieldset>

<legend>Вход в систему</legend>

<label>

Имя пользователя

<input type="text" name="username">

</label>

<label>

Пароль

<input type="text" name="password">

</label>

</fieldset>

## Атрибуты формы и полей

### disabled

Логический атрибут disabled выключает элемент управления таким образом, что он не доступен для взаимодействия или ввода.

Применение атрибута disabled к элементу <fieldset> отключит все элементы управления формы внутри группы.

<label>

Имя пользователя

<input type="text" name="username" disabled>

</label>

### placeholder

Атрибут placeholder в HTML5 предлагает подсказку или совет внутри элемента <input> или <textarea>, которая исчезает при щелчке по элементу управления или при получении фокуса. Это применяется, чтобы дать пользователям дальнейшую информацию о том, как поле формы должно быть заполнено, к примеру, использовать формат электронной почты.

<label>

Email

<input type="email" name="email-address" placeholder="name@domain.com">

</label>

Основное различие между атрибутами placeholder и value в том, что текст value остаётся на месте, когда элемент управления получает фокус, пока пользователь не удалит его вручную. Это очень удобно для предварительно заполненных данных, таких как личная информация, но не для предоставления указаний.

### required

Логический атрибут required в HTML5 утверждает, что элемент формы должен содержать значение при отправке на сервер. Если у элемента формы нет значения, будет отображаться сообщение об ошибке с просьбой пользователю заполнить обязательное поле. В настоящее время стили сообщения об ошибке контролируются браузером и не могут быть оформлены в CSS. Некорректные элементы формы, с другой стороны, могут быть стилизованы с помощью псевдоклассов :optional и :required.

Проверка также специфична для конкретного типа элемента управления. Например, элемент <input> со значением email у атрибута type потребует не только чтобы значение в поле существовало, но также чтобы это был корректный адрес электронной почты.

<label>

Email

<input type="email" name="email-address" required>

</label>