Поля для ввода текста в формы.

В саму оболочку HTML-формы между тегами <form> и </form> вставляются кнопки для отправки или очищения данных, кнопки для подтверждения выбора, переключатели, выпадающие списки и поля для ввода данных, последними как раз займемся сейчас.

Тег <input> обозначает поле для ввода текста, этот тег может принимать следующие атрибуты:

<input type="text" name="login" value="введите логин" size="20" maxlength="5">

результат:

type атрибут тега input, который указывает тип поля, в данном случае он не обязателен, с точки зрения спецификации по HTML это значение(text) по умолчанию, но лично я советую его вставлять всегда.

name это очень важный атрибут и является обязательным!!! Он является кодификатором данного поля, сценарии как раз и ждут имена полей с каким-то значением. Имя этого атрибута может начинаться только с латинской буквы и заканчиваться ними, между буквами может использоваться дефис.

value атрибут который обозначает значение поля, или точнее значение атрибута name, эти атрибуты всегда отправляются в сценарий парой, name=value или login=ваш\_логин. Этот атрибут не является обязательным, и вставляется только для того что-бы в поле отображался какой либо текст, в моем примере введите логин.

size этот атрибут не является обязательным, и устанавливает только количество символов для отображения, в моем примере 20, а ввести можно и 1000, но отображаться будут только 20. Кстати 20 задавать нет смысла т.к. это значение по умолчанию, используйте этот атрибут если вам нужно больше или меньше 20 символов для отображения.

maxlength этот атрибут служит для ограничение символов для ввода, попробуйте в моем примере стереть текст введите логин и набрать больше 5, и у вас это не получится!!! Представите что кто-то скопирует текст какой либо книги и вставит в поле где должен быть логин, это может круто увеличить нагрузку на ваш сервер, а с этим ограничителем у этого чудо-посетителя ничего не получится!!! Ограничивать 5 символами я вам вообще не рекомендую, желательно чтобы это значение было порядка 32 или даже 64!!!

Поле для ввода паролей

<input type="password" name="password" value="введите пароль" size="20" maxlength="32">

результат: Все те же атрибуты за исключением значения "password" для атрибута type. Это поле служит для ввода паролей или прочего текста который не должен видеть посторонний глаз.

Скрытое поле:

<input type="hidden" name="partner" value="идентификатор">

Это поле служит для отправки скрытых данных, Например, вы зарегистрировались на каком-то сайте по обмену эл.валют, как партнер и собираетесь немного заработать, для этого вы получаете партнерский код и ставите его у себя на сайте, а в скрытое поле ставите свой идентификатор(номер партнера) и после того как пользователь вашего сайта выберет направление перевода и нажмет кнопку "обменять" он вместе этими даными отправит ваш идентификатор и вы срубите пару копеек с этой операции. Это не единственная возможность для скрытого поля.

Поле для загрузки файла:

<input type="file" name="upload\_file" size="50" >

Результат:

При помощи данного поля можно загрузить почти любой тип файла, картинки, архивы, музыку, видео и прочие. Если вы в форме используете это поле, то занчение атрибута enctype в теге form должно быть "multipart/form-data", чтобы браузер не кодировал, а сервер на котором лежит скрипт(сценарий) не декодировал эти файлы, т.к это может просто разрушить их.

Поле для отправки комментария:

<textarea name="mesage" rows="4" cols="55" wrap="virtual"> Текст по умолчанию </textarea>

Результат:

textarea- это тег для поля , он является парным!!!

Атрибут name вам уже известен.

rows этот атрибут устанавливает количество строчек, в моем примере 4строки.

cols а этот атрибут устанавливает количество букв в одной строке.

Это и все поля которые пользователь может заполнять самостоятельно, за исключением type="hidden" .

Урок 9. Формы и поля ввода в html 5

В этой статье мы поговорим о том, как создаются формы и различные поля для ввода информации в html, какие у этих элементов есть теги и атрибуты, а также значения этих атрибутов. В html 5 добавилось довольно много полезных полей для ввода информации, не все они корректно работают в некоторых браузерах, но прогресс не стоит на месте и web – технологии развиваются, поэтому в этой статья я постараюсь наиболее полно рассказать обо всех новых инпутах и атрибутов, которые появились в HTML 5.

Создание форм в html, виды input, атрибуты тегов <form> и <input>

Формы позволяют пользователю взаимодействовать с сайтом, загружать данные или делать выбор. Это могут быть формы регистрации, обратной связи, тестирования или опроса и т.д.

Формы создаются тегом <form></form>. Данный тег должен содержать несколько обязательных атрибутов.

Обязательные атрибуты тега <form></form>:

name— название формы;

action – обработчик формы;

method – метод передачи данных:

GET – открытый метод передачи данных (данные передаются через строку браузера);

POST – скрытый метод передачи данных (Данные передаются в закрытом режиме, после соединения с сервером).

Необязательные атрибуты тега <form></form>:

novalidate – при отправке данных формы, они не будут проверяться на правильность заполненности.

autocomplete – запоминает данные, которые вы ввели в форму и при следующем вашем посещении автоматически их подставляет. Данные будут подставлены только в том случае, если форма была отправлена. Атрибут может принимать два значения:

of – выключено, стоит по умолчанию;

on – включено.

Тег <form></form>, является контейнером для своих элементов, которые создаются с помощью тега <input>. Видов инпутов в html довольно много, но все они полезные и очень часто применяются на практике. Рассмотрим виды тега <input> и атрибуты, которые можно к ним применять.

Виды инпутов

Текстовые инпуты:

<input type=«text» > – создает текстовое поле для ввода информации;

<input type=«password» > – создает поле для ввода пароля;

<input type=«email» > — поля ввода e-mail;

<input type=«url» > — инпут для ввода url – адреса;

<input type=«tel» > — поле для ввода телефона (В атрибуте pattern данного поля можно задать какое-либо регулярное выражение, например +79[0-9]{2}-[0-9]{3}-[0-9]{2}-[0-9]{2}, что будет соответствовать формату +7918-777-77-77

<input type=«search» > — поля поиска по сайту;

Инпуты, не содержащие полей для ввода текста:

<input type=«radio» > – создает радиокнопку;

<input type=«checkbox» > – создает чекбокс;

<input type=«file» > – позволяет загрузить файл. У данного инпута, есть очень полезный атрибут, который позволяет ограничить тип загружаемых файлов:

accept – устанавливает тип загружаемых данных, и может принимать следующие значения:

image/\* — загружать только файлы изображения;

audio/\* — загружать только аудио файлы;

video/\* — загружать только видео файлы.

Вместо \* можно указать через запятую типы загружаемых данных.

multiple – добавляет возможность загрузки сразу нескольких файлов.

<input type=«color» > – позволяет выбрать цвет;

<input type=«hidden» > – скрытое поле;

<input type=«range» > – создает ползунок с возможностью перемещения, к нему можно применить следующие атрибуты:

min – минимальное значение;

max – максимальное значение;

step – шаг, между числами.

<input type=«number» > – создает поле с возможностью выбора числа, атрибуты данного поля:

min – минимальное значение;

max – максимальное значение;

step – шаг, между числами.

<input type=«data» > – создает поле, с возможностью выбора даты;

<input type=«time» > – создает поле, с возможностью выбора времени;

<input type=«datetime— local» > – создает поле, с возможностью выбора одновременно и даты и времени;

<input type=«month» > — поле для выбора месяца;

<input type=«week» > — поле для выбора недели;

Инпуты-кнопки:

<input type=«submit» > - отправляет данные;

<input type=«reset» > – сбрасывает данные в форме;

<input type=«image» > – кнопка отправки данных в виде картинки. У этого инпута есть ряд обязательных атрибутов:

name – имя инпута (данный атрибут обязателен для всех видов инпутов);

src – путь к изображению;

alt – альтернативный текст, который будет написан на кнопке, в случае, если изображение не загрузится.

width – ширина (необязательный атрибут);

height – высота (необязательный атрибут);

<input type=«button» > – просто кнопка (используется при управлении скриптами).

Для того чтобы сделать текст инпута формы активным, необходимо воспользоваться тегом <label></label>, тогда, по щелчку на текст, инпут будет становиться активным для ввода информации. Например:1 <label>Введите пароль: <input type="password" name="password" required></label>

Для лучшего понимания, предлагаю создать форму авторизации. Для этого нам внутри формы понадобится два инпута с полем ввода для текста и пароля. Код такой формы авторизации будет выглядеть:

<!DOCTYPE HTML>

<html>

<head>

<meta charset="utf-8" />

<title>Урок 9. Формы в HTML</title>

</head>

<body>

<form action="form.php" method="POST" name="form" >

<label>Введите ваше имя: <br>

<input type="text" name="name" required size="40" maxlength="25">

</label> <br>

<label>Введите ваш e-mail: <br>

<input type="email" name="email" required size="40" maxlength="25">

</label>

</form>

</body>

</html>

Чтобы наша форма имела приятный вид, были расставлены теги переноса строки <br>, а для того, чтобы по щелчку на текст, поля ввода становились активными, был применен тег <label></label>. Атрибут required делает наши поля обязательными к заполнению. В итоге, наша форма авторизации выглядит вот так:

Атрибуты тега <input>:

Обязательные атрибуты:

name – имя инпута;

type – тип инпута.

Необязательные атрибуты:

maxlength – количество символов, которые можно вводить в инпут.

size – длинна поля инпута;

value – Значение инпута по умолчанию;

disabled – блокирует определенный элемент;

readonly – позволяет только читать, но удалить или что-то ввести нельзя.

tabindex – определяет порядок переключения между элементами формы. В качестве значения указываем число – порядковый номер элемента формы.

required –делает поле для ввода обязательным;

pattern – добавляет регулярное выражение для поля ввода;

autofocus – автоматически ставит курсор на поле, помеченное данным атрибутов, в значение необходимо указать autofocus;

placeholder – подсказка в поле, которая исчезает при активации поля.

autocomplete – автозаполнение инпута (имеет два значения – on и off);

spellcheck – добавляет проверку орфографии в поле ввода текста:

true – включено;

false – выключено;

default – проверка орфографии будет зависеть от браузера.

Как добавить всплывающие подсказки в поле ввода текста?

В html 5 появился такой дескриптор, как <datalist></datalist>, который является контейнером для различных опций, задаваемых тегом <option>. В теге <option> достаточно указать атрибут value с необходимым нам значением. Можно создавать неограниченное количество опций с разными значениями.

Для того чтобы подключить существующий <datalist></datalist> к тому или иному инпуту, необходимо указать в теге <datalist> его идентификатор Id, а инпуту назначить атрибут list, c именем этого идентификатора.

Добавим к полю поиска несколько подсказок, код будет выглядеть так:

<!DOCTYPE HTML>

<html>

<head>

<meta charset="utf-8" />

<title>Урок 9. Формы в HTML</title>

</head>

<body>

<form action="form.php" method="POST" name="form" >

<input list="myEmailList" type="search" name="searc" size="40" maxlength="25">

<datalist id="myEmailList">

<option value="Игры" >

<option value="Музыка" >

<option value="Фильмы" >

<option value="Программы" >

</datalist>

</form>

</body>

</html>

На скриншоте можно увидеть, что подсказки действительно появились:

Как сделать радиокнопку в html?

Радиокнопки создаются тегом <input type="radio">. Все радиокнопки, объединенные в одну группу, должны иметь одинаковые имена. Значение радиокнопки указывается в атрибуте value. Также, с помощью атрибута checked, можно задать радиокнопку, выбранную по умолчанию.

Давайте рассмотрим простой пример. Создадим группу радиокнопок с выбором марки автомобиля. Пусть, например, необходимо выбрать из трех видов производителей автомобилей, тогда код группы таких радиокнопок будет выглядеть:

<!DOCTYPE HTML>

<html>

<head>

<meta charset="utf-8" />

<title>Урок 9. Формы в HTML</title>

</head>

<body>

<form action="form.php" method="POST" name="form" >

<p>Вы владелец автомобиля марки:</p>

<input type="radio" name="car" value="vaz" checked> ВАЗ <br>

<input type="radio" name="car" value="bmw"> BMW <br>

<input type="radio" name="car" value="toyota"> TOYOTA <br>

</form>

</body>

</html>

А результат работы такого кода выглядит:

Обратите внимание, для первого элемента формы, был задан атрибут checked, на который и установлена по умолчанию наша радиокнопка. Как видно из примера, радиокнопки позволяют выбрать лишь один вариант, но вот если встает необходимость выбора сразу нескольких вариантов, то можно воспользоваться чекбоксами.

Как создать чекбокс в html?

Чекбоксы создаются тегом <input >. Внутри инпута необходимо указать его тип (checkbox) и значение атрибута name.

Параметр value может принимать два значения:

on – включено;

of – выключено.

Создадим группу из пяти чекбоксов с выбором языков. Код будет выглядеть:

<!DOCTYPE HTML>

<html>

<head>

<meta charset="utf-8" />

<title>Урок 9. Формы в HTML</title>

</head>

<body>

<form action="form.php" method="POST" name="form" >

<p>Выберете языки, которыми вы владеете:</p>

<input type="checkbox" name="Russian" checked> Русский <br>

<input type="checkbox" name="English" > Английский <br>

<input type="checkbox" name="France" > Французcкий <br>

<input type="checkbox" name="Japan" > Японский <br>

<input type="checkbox" name="China" > Китайский <br>

</form>

</body>

</html>

Результат работы кода, ниже на скриншоте:

Но какой смысл создавать чекбоксы, если данные из них мы не можем отправить, а для этих целей используются кнопки. Существует несколько видов кнопок:

submit – отправляет данные;

image – кнопка отправки данных в виде картинки;

reset – кнопка сброса;

file – позволяет загрузить файл;

button – кнопка какого – либо действия, которое задается с помощью скрипта.

В значение value, указывается текст, который будет написан на кнопке.

Добавим к нашей форме регистрации кнопку отправки данных и сброса. Наш код примет вид:

<!DOCTYPE HTML>

<html>

<head>

<meta charset="utf-8" />

<title>Урок 9. Формы в HTML</title>

</head>

<body>

<form action="form.php" method="POST" name="form" >

<label>Введите ваше имя: <br>

<input type="text" name="name" required size="40" maxlength="25">

</label> <br>

<label>Введите ваш e-mail: <br>

<input type="email" name="email" required size="40" maxlength="25">

</label>

<br>

<!-- Кнопки отправки и сброса -->

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

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

</form>

</body>

</html>

На скриншоте видно, что кнопки действительно появились, но кнопка отправки пока не работает потому, что мы еще не добавили обработчик, который указали в атрибуте action. Позже мы к этому вернемся.

Довольно часто на сайтах и блогах применяется выпадающий список с возможностью выбора одного или нескольких элементов списка. Давайте рассмотрим, как создать раскрывающийся список выбора.

Как сделать раскрывающийся список выбора в html?

Чтобы создать такой список, необходимо вовнутрь тега <select></select>, поместить элементы списка, с помощью тега <option></option>. Таким образом, тег <select></select> является контейнером для элементов раскрывающегося списка и для него необходимо указать атрибут name. В нашем контейнере прописываем нужное количество элементов списка тегом <option></option>, в котором указывает атрибут value со значением элемента.

Для группировки элементов выпадающего списка применяется тег <optgroup></optgroup>, в котором необходимо указать атрибут label со значением группируемых элементов.

Для того чтобы сделать список по умолчанию развернутым необходимо задать атрибуты multiple и size, которые указываются в атрибуте <select></select> , где в size указывается число элементов, которые мы можем выбрать одновременно

Для более наглядного понимания, давайте добавим к нашей форме регистрации возможность выбора даты рождения. Код будет выглядеть так:

<!DOCTYPE HTML>

<html>

<head>

<meta charset="utf-8" />

<title>Урок 9. Формы в HTML</title>

</head>

<body>

<!-- Форма регистрации -->

<form action="formOne.php" method="POST" name="formOne" >

<!-- Поле ввода имени -->

<label>Введите ваше имя: <br>

<input type="text" name="name" size="40" maxlength="25" reuiared>

</label> <br>

<!-- Поле ввода пароля -->

<label>Введите ваш пароль: <br>

<input type="password" name="password" size="40" maxlength="25" reuiared >

</label> <br>

<!-- Раскрывающийся список, с выбором даты рождения -->

<select name="happy" >

<optgroup label="День">

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

<option value="7"> 7 </option>

<option value="15"> 15 </option>

</optgroup>

<optgroup label="Месяц">

<option value="Апрель"> Апрель </option>

<option value="Май"> Май </option>

<option value="Январь"> Январь </option>

</optgroup>

<optgroup label="Год">

<option value="1987"> 1987 </option>

<option value="1965"> 1965 </option>

<option value="2014"> 2014 </option>

</optgroup>

</select>

<br>

<!-- Кнопки отправки и сброса -->

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

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

</form>

</body>

</html>

Результат, можно посмотреть на скриншоте:

Группировать можно не только элементы списков, но и элементы формы, как это сделать разберем ниже.

Как группировать элементы формы в html?

Для того, что сгруппировать какие-либо элементы формы, необходимо поместить их в тег <fieldset></fieldset>, а название, объединенной группы элементов формы, необходимо указать в теге <legend></legend>, который следует сразу же за тегом <fieldset></fieldset>.

Предлагаю объединить элементы ввода логина и пароля в одну группу и назвать «Авторизация». Код будет выглядеть:

<!DOCTYPE HTML>

<html>

<head>

<meta charset="utf-8" />

<title>Урок 9. Формы в HTML</title>

</head>

<body>

<form action="form.php" method="POST" name="form" >

<fieldset>

<legend>АВТОРИЗАЦИЯ</legend>

<label>Введите ваше имя: <br>

<input type="text" name="name" required size="40" maxlength="25">

</label> <br>

<label>Введите ваш e-mail: <br>

<input type="email" name="email" required size="40" maxlength="25">

</label>

</fieldset>

<br>

<!-- Кнопки отправки и сброса -->

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

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

</form>

</body>

</html>

На скриншоте видно, что поле ввода логина и пароля объединились в логическую группу и выделяются рамкой:

Одной из наиболее часто используемых функций при создании форм, является отправка через формы письма на почту, но в этом случае, необходимо создать поле для ввода текста. Как это сделать обсудим ниже.

Как добавить поле для ввода текста?

Для создания текстовой области необходимо воспользоваться тегом <textarea></textarea>, у которого есть несколько атрибутов:

Атрибуты тега <textarea></textarea>:

name – имя текстовой области;

rows – высота области в символах;

cols – ширина в символах.

disable – выключить текстовое поле;

wrap – данный атрибут позволяет сохранить форматирование текста, введенного в текстовое поле.

soft – форматирование не сохраняется, перевода на новую строку не будет;

hard – форматирование текста будет сохранено;

off – выключить переносы на новую строку.

Создадим простейшую форму отправки письма на почту и добавим к ней обработчик. Итак, код нашей формы будет выглядеть:

<!DOCTYPE HTML>

<html>

<head>

<meta charset="utf-8" />

<title>Урок 9. Формы в HTML</title>

</head>

<body>

<form action="form.php" method="POST" name="form" >

<label>Введите ваше имя: <br>

<input type="text" name="name" required size="40" maxlength="25">

</label> <br>

<label>Введите ваш e-mail: <br>

<input type="email" name="email" required size="40" maxlength="25">

</label>

<br>

<textarea name="text" rows="10" cols="40">Введите сюда текст, который вы хотите отправить и нажмите - Отправить</textarea>

<br>

<!-- Кнопки отправки и сброса -->

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

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

</form>

</body>

</html>

Результат работы этого кода ниже:

Но пока наша форма отправки письма на почту не работает, так как для ее полноценной работы необходим php скрипт, который будет выполнять всю работу на стороне сервера. Это конечно не входит в нашу сегодняшнюю статью, но все же давайте кратко рассмотрим, как выглядит этот самый обработчик.

Обработчик формы отправки письма на e-mail

Стоит заметить, для того, чтобы обработчик мог функционировать, у вас на компьютере должен быть установлен виртуальный сервер, или используйте код, который мы рассмотрим ниже уже на реальном сервере.

Итак, создадим обработчик формы отправки на e-mail, который указан в атрибуте action нашей формы, и сохраним этот файл с расширение php.

Код обработчика отправки формы на e-mail:1

<?php header('Content-Type: text/html; charset=UTF-8'); ?>

<?php

$result=mail("nemyzhik@yandex.ru", "Анкета с сайта", "С сайта была получена анкета с данными: \nИмя: $\_POST[name] \nE-mail: $\_POST[email] \nТекст письма: $\_POST[txt]");

if ($result) {

echo "Письмо отправлено";

}

Else {

Echo "Письмо не отправлено" ;

}

?>

Это самый простой обработчик отправки письма на e-mail, который только может существовать, но все же разберем его немного подробнее.

Первая строка задает кодировку UTF-8, для того, чтобы вам на почту не приходили неизвестные символы вместо русских букв. Обязательным условием является то, чтобы эта строчка была первой и перед ней не было пробелов и других символов. А также сохраните обработчик в кодировке utf-8 без bom.

Во-второй строчке мы создаем переменную $result, в которую помещаем функцию отправки письма mail со всеми данными, введенными в наши поля.

И в конце добавляем проверку, если письмо отправится, то получим сообщение «Письмо отправлено», если нет, то «Письмо не отправлено».

Видеоверсия урока

Пожалуй, это вся информация, которую вы должны знать о формах в html 5, полях ввода, атрибутах форм и их значениях. Статья получилась довольно объемной и, на первый взгляд, тяжелой для восприятия. Но вы всегда можете вернуться на страницы блога nemyzhik.ru и освежить информацию. На этом я с вами прощаюсь, искренне надеюсь, что данная статья будет для вас полезной, и вы освоите создание форм и полей ввода в html.

ПОХОЖИЕ

SEO

ОПТИМИЗАЦИЯ САЙТА

ОСНОВЫ HTML

УРОК

HTML-форма и ее основные элементы

Опубликовано в Верстка admin Ноябрь 15, 2016

HTML—форма – это специальный модуль, предназначенный для ввода и последующей отправки на обработку каких—то данных, к примеру, логина и пароля, информации о пользователе, сообщения и т.д. Чаще всего используются следующие виды форм – регистрация, авторизация, отправка сообщения, отправка комментария, отправка письма на почту, анкета.

Вставка формы осуществляется напрямую в HTML—код страницы. Главный элемент формы называется <form>. Уже внутрь него добавляются все остальные элементы – текстовые поля, «чекбоксы», переключатели и т.д. У элемента <form> имеется несколько атрибутов, один из которых является обязательным. Он называется action. В action указывается, где именно будет приниматься и обрабатываться информация, переданная посредством формы. Как правило, обработка происходит в стороннем PHP—файле. Пример использования атрибута– action=»obrabotchik.php«. Атрибут method позволяет задать метод передачи информации. По умолчанию (если не прописывать атрибут) будет указан метод GET. В данном случае информация передается напрямую через URL—адрес. Для каждого элемента формы будет создана пара следующего вида – «имя элемента = значение, которое в нем лежит». Все эти пары, разделенные знаком «амперсанд» будут перечислены в адресной строке. Если прописать method=»POST» (регистр не важен), то данные будут передаваться не через URL, а через тело запроса (в скрытом режиме). В большинстве случаев используют именно POST. Пример создания формы:

1

2

3 <form action="obrabotka.php" method="POST">

/\*\*\*\*\*ПОЛЯ ВВОДА\*\*\*\*\*\*\*\*/

</form>

Атрибуты HTML-формы

Атрибут name — позволяет присвоить форме уникальное имя. Это нужно в том случае, если на одной странице размещено несколько форм.

Атрибут enctype — позволяет задать способ кодирования информации из формы. Обычно его заполняют только тогда, когда при помощи формы происходит пересылка файлов. Вообще, здесь может быть три варианта — enctype= application/x—www—form—urlencoded | multipart/form—data | text/plain. Первый вариант application/x—www—form—urlencoded говорит о том, что происходит пересылка закодированного текста. Третий вариант text/plain используют для пересылки текста в формате «как есть». Второй вариант multipart/form—data используют для файлов.

Атрибут формы autocomplete позволяет включить так называемое автозаполнение полей.

Поля ввода в HTML — форме

Теперь расскажем о том, какие элементы может содержать в себе форма. Прежде всего, это разного рода input’ы – элементы, предназначенные для ввода данных.

Текстовое поле — type=»text»

Самый распространенный input – обычное однострочное текстовое поле. В коде оно может выглядеть следующим образом:

1 <input type="text" name="t" maxlength="10" value="Иван">

В атрибуте type, как вы поняли, указывается тип поля. Атрибут name – это уникальное имя поля (оно нужно для последующей обработки данных). Атрибут maxlength задает максимальную длину вводимого текста. Value – это значение поля по умолчанию (после загрузки страницы в этом поле сразу же будет прописано слово «Иван»).

Поле для пароля — type=»password»

Еще один популярный input – поле для ввода пароля. Выглядит оно вот так:

1 <input type="password" name="p">

Ключевая особенность – вводимые символы будут отображаться как звездочки.

Чекбокс — type=»checkbox»

Следующий элемент – чекбокс. В коде он может выглядеть примерно так:

1 <input type="checkbox" id="ch" name="c" value="1" checked>

Атрибут type здесь также задает тип. ID – это уникальный идентификатор, может использоваться для привязки к чекбоксу CSS—правил или текстовой надписи. Value – это значение, посылаемое на обработку в том случае, если в чекбоксе поставлена галочка. Атрибут checked автоматически ставит галочку в чекбоксе сразу при загрузке страницы. У checked не нужно указывать значение, так как оно совпадает с именем атрибута.

Надпись над элементом — тег label

Следующий элемент – текстовая надпись (элемент label). Как правило, текст привязывают к какому—то уже существующему input’у. В этом случае надпись может выглядеть вот так:

1 <label for="ch">Текст для чекбокса</label>

Атрибут for осуществляет непосредственную привязку текста к элементу с id=»ch» (в нашем случае это чекбокс).

Переключатель — type=»radio»

Еще один интересный элемент – это радиопереключатель. Обычно его вставляют в форму, что бы предоставить пользователям возможность выбора из нескольких равнозначных вариантов. Конструкция, приведенная ниже, создаст радиопереключатель из двух input’ов.

1

2 <input type="radio" name="n" value="qwe" checked>

<input type="radio" name="n" value="qwe2">

Атрибут type задает тип input’а. Name – это имя. Все input’ы в пределах одного переключателя должны иметь одно имя, иначе форма не свяжет их друг с другом. Value –это значение, передающееся при выборе того или иного элемента. Атрибут checked позволяет установить один и только один из элементов в включенное по умолчанию состояние.

Выпадающий список — тег select

Элемент формы select позволяет создать выпадающий список. У него есть несколько атрибутов. Name задает уникальное имя. Size задает количество элементов, отображающихся в списке одновременно (при загрузке страницы). По умолчанию size=1. Атрибут без значения multiple позволяет выбирать сразу несколько элементов, предварительно зажав клавишу Ctrl. Чтобы создать непосредственные элементы списка, используется тег option. Текстовое содержимое данного тега будет отображаться в качестве элемента. При указании атрибута value внутри options на обработку будет передано его значение. Атрибут selected позволяет сделать выбранный элемент списка текущим (по аналогии с checked он будет выделен сразу при загрузке страницы). Несколько элементов можно объединить в группу, присвоив ей какое—то название. Делается это посредством тега optgroup, внутрь которого вставляются нужные option. Атрибут label задает группе название. Ниже приведен пример сгруппированного списка.

<select name="s" size="3" multiple>

<optgroup label="Список">

<option value="1">1 элемент</option>

<option value="2">2 элемент</option>

<option value="3" selected>3 элемент</option>

<option value="4">4 элемент</option>

</optgroup>

</select>

Многострочное текстовое поле — тег textarea

Довольно часто в формах используется многострочное текстовое поле. Обычно это делается при отправке почтовых сообщений и комментариев. Элемент, создающий такое поле, называется textarea. Вся конструкция может выглядеть примерно так:

<textarea maxlength="1000" name="area" rows="5" cols="30">

</textarea>

Атрибут maxlength задает максимальное количество символов, вводимых в поле. Атрибуты rows и cols – это число строк и символов в строке соответственно. Стоит помнить, что размеры формы не являются фиксированными – ее всегда можно расширить. Однако при загрузке страницы она будет появляться именно с размерами, указанными в rows и cols.

Кнопки в HTML — форме

Последние элементы, о которых мы поговорим – это кнопки. Типичная кнопка отправки формы может выглядеть так:

1 <input type="submit" value="Послать">

В атрибуте value указывается надпись, которая будет отображаться на кнопке. Что касается type, то здесь может быть несколько вариантов. Submit создает кнопку подтверждения и отправки данных. Reset создает кнопку сброса всех введенных полей в изначальное состояние. Button создает кастомную кнопку без привязанного к ней действия по умолчанию (впоследствии можно будет навесить на нее какой—нибудь скрипт). Image также создает кастомную кнопку, у которой вместо текста будет использоваться изображение. Путь к изображению необходимо указать в атрибуте src.

Текстовые поля форм в HTML. Как осуществить настройку ввода данных?

учебные курсы /

html для начинающих /

текстовые поля форм

Поле ввода однострочного текста

HTML определяет два вида полей, отвечающих за ввод текстовых данных. Элемент <textarea> используется для написания многострочного текста, а <input> принимает лаконичную информацию, которая может вместиться в одной строке (Ф.И.О, адрес юридического лица, номер телефона и т.п.).

Эти элементы имеют много дополнительных атрибутов, которые изменяют способы взаимодействия с полем, и накладывают ограничения на вводимый текст. Элемент <textarea> станет предметом рассмотрения следующей статьи, а сейчас мы подробно изучим поле <input> и его возможности по обработке однострочных текстовых данных.

Элемент <input> является самым многозадачным в HTML. Различные значения атрибута type кардинальным образом меняют его поведение. Он может отвечать за выбор и загрузку файлов, ввод даты, выбор цвета из палитры, и многое другое. Мы рассмотрим только те значения, которые преобразуют его в поле для ввода текста.

Список значений атрибута type:

text — Основной тип текстового поля. Является универсальным и, по сути, можно использовать только его, не обращая внимание на остальные.

password — Ввод пароля. Все символы в таком поле будут отображены с помощью звездочек «\*». Это необходимо для предотвращения подглядывания за набираемым текстом.

url — Ввод универсального указателя ресурса (адрес сайта). Например, https://coder-booster.ru.

email — Адрес электронной почты.

search — Текст, участвующий в поиске.

tel — Поле телефонного номера.

<!DOCTYPE html>

<html>

<head>

<meta charset="utf-8">

<title>Пример создания текстовых полей формы</title>

</head>

<body>

<form>

<!-- текстовые поля ввода -->

<input type="email" placeholder="Логин">

<input type="password" placeholder="Пароль">

<!-- кнопка отправки формы -->

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

</form>

</body>

</html>

В чем различие типов полей

HTML определяет особую семантику для каждого типа, приведенного выше. В браузере эти поля отображаются одинаково. Разница будет ощутима при взаимодействии с ними на мобильных устройствах.

У современных смартфонов отсутствуют буквенные клавиши. При нажатии на текстовом поле ввода появляется виджет клавиатуры. В зависимости от атрибута type элемента <input>, содержимое этого виджета будет отличаться. К примеру, нажав на поле ввода адреса электронной почты, виджет переключится на английский язык и будет содержать все символы, участвующие в формировании стандартного Email-адреса.

Блокирование содержимого поля

Элемент <input> можно заблокировать для редактирования. В HTML есть атрибут disabled. Его использование окрасит поле ввода в серый цвет, а при отправке формы данные не будут включены в тело запроса.

Атрибут readonly обладает схожим поведением. Его добавление не вызовет закрашивание поля серым цветом, но текстовое содержимое станет недоступным для редактирования. В отличие от атрибута diasbled, данные из такого текстового поля будут включены в тело запроса при отправке.

<!DOCTYPE html>

<html>

<head>

<meta charset="utf-8">

<title>Пример блокировки полей от редактирования</title>

</head>

<body>

<form>

<!-- текстовые поля ввода -->

<label>

<span>Ваш любимый цвет:</span><br>

<input disabled type="text" value="Зеленый">

</label><br><br>

<label>

<span>Направление музыки:</span><br>

<input readonly type="text" value="Post-Rock">

</label><br><br>

<!-- кнопка отправки формы -->

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

</form>

</body>

</HTML>

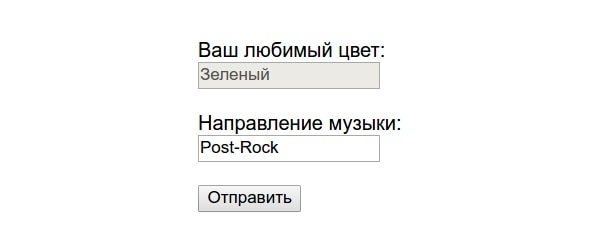


рис 1. Пример отображения заблокированных полей ввода

Значения и подсказки текстовых полей

Поля могут иметь некоторые предустановленные значения. Совсем не обязательно дожидаться ввода текста от пользователя. Для установки содержимого необходимо использовать атрибут value. Если пользователь не изменит предустановленные значения, при отправке они будут включены в тело запроса как есть.

HTML также позволяет помочь пользователю с помощью подсказок. Значение атрибута placeholder будет показано светло серым цветом внутри поля ввода, если оно является пустым. Когда пользователь начнет добавление текста – подсказка исчезнет.

<!DOCTYPE html>

<html>

<head>

<meta charset="utf-8">

<title>Пример установки значений и подсказок</title>

</head>

<body>

<form>

<!-- текстовые поля ввода -->

<label>

<span>Желаемая должность:</span><br>

<input type="text" value="Веб-дизайнер">

</label><br><br>

<label>

<span>Контактные данные:</span><br>

<input type="tel" placeholder="Телефон">

<input type="email" placeholder="Email">

</label><br><br>

<!-- кнопка отправки формы -->

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

</form>

</body>

</html>

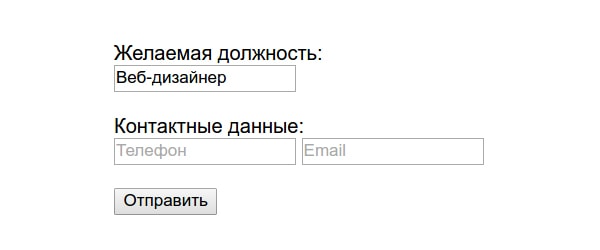


рис 2. Показ предустановленных значений и подсказок

Процесс отправки и проверка данных

Атрибут name указывает уникальное имя текстового поля. Он является не обязательным, но в случае отсутствия, значение не будет включено в тело запроса при отправке формы. Как правило, он задается короткой фразой, характеризующей содержимое.

HTML позволяет осуществлять проверку введенных данных. Если они не будут соответствовать требованиям, при нажатии кнопки отправки, форма выведет сообщения об ошибках и потребует исправлений.

Специальный атрибут required обязывает поле иметь значение. Если пользователь оставит его пустым, отправка формы не состоится. Существует возможность и более сложных проверок текстовых полей. Атрибут pattern указывает в своем значении регулярное выражение, по правилам которого будет производиться проверка данных. Значение «[0-9]{5,10}» соответствует только числам, содержащим от пяти до десяти цифр.

Вы можете ограничить количество вводимых символов с помощью атрибута maxlength. В его значении должно находиться число. При попытке добавить большее количество символов в текстовое поле, их ввод будет заблокирован.

<!DOCTYPE html>

<html>

<head>

<meta charset="utf-8">

<title>Проверка данных при отправке формы</title>

</head>

<body>

<form>

<!-- текстовые поля ввода -->

<label>

<span>Имя:</span><br>

<input required maxlength="60" name="username" type="text">

</label><br><br>

<label>

<span>Год родения:</span><br>

<input required pattern="[0-9]{4}" name="birth" type="text">

</label><br><br>

<!-- кнопка отправки формы -->

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

</form>

</body>

</html>

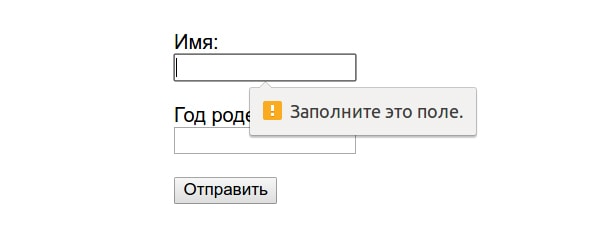


рис 3. Сообщение о неправильно заполненном поле в браузере Google Chrome

Заметка

Атрибуты required и pattern не работают в браузере Safari. Если вы хотите выполнять полную проверку данных, которая будет работать во всех браузерах, нужно использовать JavaScript.« Управление отправкой данных формы Создание полей ввода многострочного тес