

top

КОМПЬЮТЕРНАЯ
АКАДЕМИЯ

HyperText Markup Language HTML&CSS

Урок 10

Формы

Оглавление

Формы.....	4
Домашнее задание.....	37

Материалы урока прикреплены к данному PDF-файлу. Для доступа к материалам, урок необходимо открыть в программе [Adobe Acrobat Reader](#).

Формы

Современный человек тратит большое количество времени на работу в Интернете. Он заходит на различные сайты, покупает товары, ищет информацию. Очень часто для решения своих задач пользователь использует инструмент под названием — форма. Это слово может звучать незнакомо для вас, но у вас уже точно есть опыт работы с формами.

Например, формы помогают вам искать информацию. Типичные примеры поисковых форм:

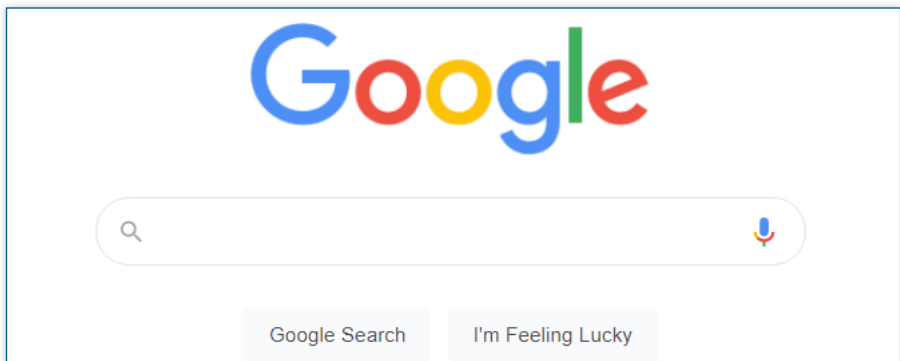
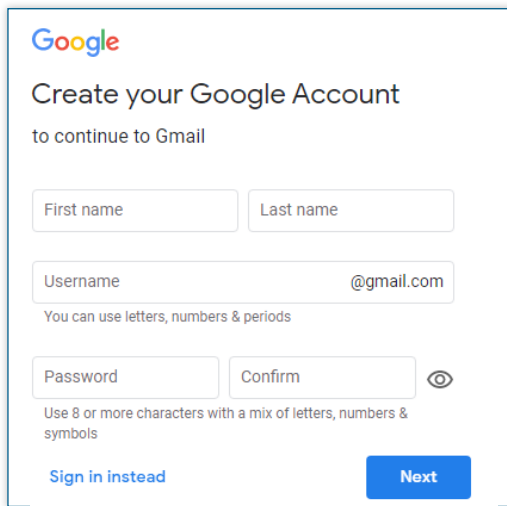


Рисунок 1



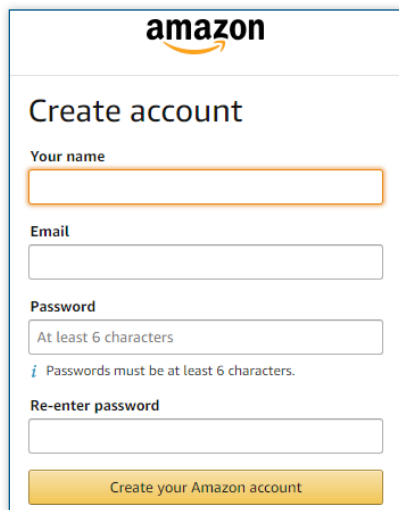
Рисунок 2

Формы используются не только для поиска информации. При регистрации на сайтах вы могли видеть подобные формы:



The image shows the Google Account creation interface. At the top is the Google logo. Below it, the text 'Create your Google Account' is followed by 'to continue to Gmail'. The form contains several input fields: 'First name', 'Last name', 'Username' (with a placeholder '@gmail.com'), 'Password', and 'Confirm'. A small eye icon is next to the 'Confirm' field. Below the password fields, there is a note: 'Use 8 or more characters with a mix of letters, numbers & symbols'. At the bottom left is a link 'Sign in instead' and at the bottom right is a blue 'Next' button.

Рисунок 3



The image shows the Amazon account creation interface. At the top is the Amazon logo. Below it, the text 'Create account' is displayed. The form contains several input fields: 'Your name', 'Email', 'Password' (with a placeholder 'At least 6 characters'), and 'Re-enter password'. Below the 'Password' field, there is a note: 'i Passwords must be at least 6 characters.' At the bottom is a yellow button labeled 'Create your Amazon account'.

Рисунок 4

Что же такое форма? Это набор элементов управления, который позволяет веб-приложению взаимодействовать с пользователем. Например, если мы ищем информацию, форма предоставляет возможность ввести слово для поиска. Если мы хотим создать аккаунт в веб-магазине, форма регистрации позволяет нам предоставить информацию о себе.

Как же встроить форму внутрь HTML-страницы? Для этого используется тег `<form>`. При использовании `form` вам требуется указать, как открывающий, так и закрывающий тег. Элементы управления должны находиться между открывающим и закрывающим тегом `<form>`. Общая структура:

```
<form>  
    элементы управления  
</form>
```

У тега `form` есть свой набор атрибутов. Мы познакомимся с ними чуть позже.

Для создания большого количества элементов управления используется тег `<input>`. Какой конкретно элемент управления нужно создать указывается с помощью атрибута `type` этого тега. Есть также элементы управления, которые создаются с помощью других тегов. Об этом мы также поговорим позднее.

Создадим первый пример по работе с формами. Наша страница будет отображать поисковую форму с текстовым полем и кнопкой для отправки данных на сервер.

```

<!DOCTYPE html>

<html lang="en">
  <head>
    <meta charset="utf-8" />
    <title>First form</title>
  </head>

  <body>
    <h2>Let's search something</h2>
    <form>
      <input type="text" />
      <input type="submit" value="Search" />
    </form>
  </body>
</html>

```

Отображение в браузере:



The image shows a browser window with a blue border. Inside, the text "Let's search something" is displayed in a large, bold, black serif font. Below this text is a white rectangular text input field with a thin grey border. To the right of the input field is a grey rectangular button with the word "Search" in black text.

Рисунок 5

Для создания формы мы используем тег `<form>`.

```

<form>
  <input type="text" />
  <input type="submit" value="Search" />
</form>

```

В коде мы создаём два элемента управления: текстовое поле и кнопку для отправки информации. Текстовое

поле создано с помощью тега `<input>`. Атрибут `type` со значением `text` показывает, что мы хотим создать именно текстовое поле. Кнопка для отправки данных на сервер также была создана с помощью `input`, но в качестве `type` было указано значение `submit`. С помощью атрибута `value` мы задали надпись на кнопке.

Кнопки типа `submit` всегда используются для отправки данных на сервер.

Попробуйте ввести значение в текстовое поле и нажать кнопку `Search`. Что у вас произошло? Страница перегружилась и значение из текстового поля пропало. Почему так произошло? Попробуем разобраться.

Как работает стандартный механизм отправки данных при нажатии на кнопку типа `submit`?

1. Браузер отправляет, полученные от пользователя данные на сервер
2. Сервер получает данные, формирует ответ и отправляет его клиенту (браузеру в нашем случае)
3. Браузер получает ответ сервера и отображает его на экран пользователю

В нашем примере нет никакого взаимодействия с сервером. Мы нигде не указывали куда необходимо отправить данные. Именно поэтому страница просто обновляется. Код нашего проекта находится в папке *First form*.

Усложним немного наш пример. Добавим ещё одну кнопку и настроим некоторые атрибуты `form`.

```
<!DOCTYPE html>
<html lang="en">
  <head>
```

```

<meta charset="utf-8" />
<title>Simple search form</title>
</head>

<body>
  <h2>Let's search something</h2>
  <form action="/search" method="POST">
    <input type="text" /><br />
    <input type="submit" value="Search" />
    <input type="reset" value="Clear" />
  </form>
</body>
</html>

```

Результат работы:



The screenshot shows a web browser window with a title bar. Inside, the page has a heading "Let's search something" in a large, bold, serif font. Below the heading is a simple search form. It consists of a single-line text input field. Underneath the input field are two buttons: "Search" and "Clear". Both buttons have a light gray background and a thin border. The "Clear" button is slightly to the right of the "Search" button.

Рисунок 6

Попробуйте ввести значение в текстовое поле и нажать на кнопку **Clear**. Значение из текстового поля будет стерто. При выполнении очистки страница не будет отсылаться на сервер.

Проанализируем нововведения в коде. У тега **form** появились атрибуты.

```

<form action="/search" method="POST">

```

В атрибуте **action** нужно указать путь на сервере, который будет обрабатывать данные формы. Если атрибут

action не указан, в качестве обработчика на сервере выбирается текущая страница. Значение для атрибута **action** обычно определяет программист, который занимается серверной частью веб-приложения.

В атрибуте **method** нужно указать какой метод для отправки данных на сервер нужно использовать. Есть два варианта: **GET** и **POST**. Рассмотрим каждый вариант отправки данных. Начнем с **GET**.

Когда в форме указан **GET**, как метод отправки, введенные данные отображаются в адресной строке браузера. Например:

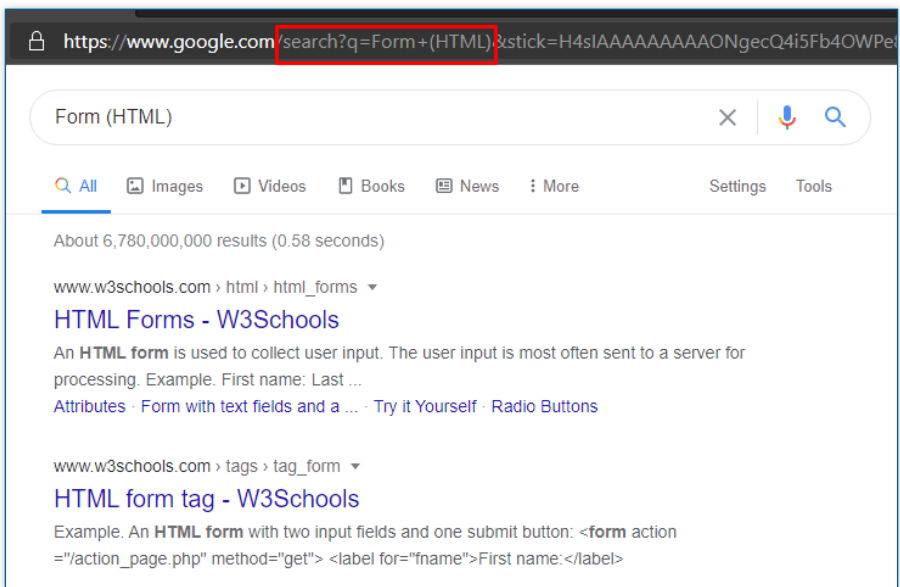


Рисунок 7

GET обычно указан, как метод отправки для поисковых систем. Если в качестве **method** указан **POST**, данные отсылаются в теле HTTP запроса и не отображаются

в адресной строке. Отсюда можно сделать вывод, что не стоит использовать **GET** для отсылки важных данных, потому что они будут видны в URL браузера и любой человек подошедший к вашему компьютеру сможет увидеть их.

Важные факты о **GET**:

- Размер URL-строки в вашем браузере ограничен (примерно 3000 символов). Это значит, что у нас есть ограничение на объём, передаваемых данных через **GET**;
- Если пользователь вашего сервиса хочет сделать закладку по URL, лучше использовать **GET**. Например, создание закладки на конкретный поисковый запрос;
- **GET** лучше использовать для несекретных данных

Важные факты о **POST**:

- Нет ограничения по размеру передаваемых данных
- Если форма использует **POST**, пользователь не сможет создать закладку на полученный результат

Решение о конкретном методе нужно принимать, внимательно проанализировав все факторы.

В нашем примере мы указали значение **POST** для **method**. Если значение для **method** не указать, по умолчанию стоит **GET**.

Кроме того, мы ввели в нашем примере новый вид элемента управления.

```
<input type="reset" value="Clear" />
```

Кнопка **reset** используется для сброса значений, введенных в элементы управления формы.

Код проекта находится в папке *Simple search form*.

Расширим спектр известных нам элементов управления. Для этого мы создадим регистрационную форму с большим количеством полей. Внешний вид примера:



We need your opinion!

Name:

Email:

Message:

Рисунок 8

Мы просим пользователя ввести: имя, email, сообщение (многострочное, текстовое поле). В этом примере мы не форматировали внешний вид, чтобы не усложнять пример. В следующих примерах мы дойдем до внешнего вида наших форм.

Обратите внимание, если кликнуть левой кнопкой мышки по надписи слева от элемента управления, элемент управления получает фокус. Попробуем кликнуть на **Name**:



We need your opinion!

Name:

Email:

Message:

Рисунок 9

В фокусе текстовое поле, справа от надписи **Name**.
Код нашей страницы:

```
<!DOCTYPE html>

<html lang="en">
  <head>
    <meta charset="utf-8" />
    <title>Send message to us</title>
  </head>

  <body>
    <h3>We need your opinion!</h3>
    <form action="/search" method="POST">
      <div>
        <label for="uname">Name:</label>
        <input type="text" id="uname" name="userName" />
      </div>
      <div>
        <label for="email">Email:</label>
        <input type="text" id="email" name="userEmail" />
      </div>
      <div>
        <label for="message">Message:</label>
        <textarea id="message" name="userMessage">
        </textarea>
      </div>
      <div>
        <input type="submit" value="Send my message" />
      </div>
    </form>
  </body>
</html>
```

Какие новые элементы есть в нашей форме?

Первый элемент — это надпись. Её мы создаем тегом **label**. Он используется для создания надписи. С помощью

атрибута `for` мы указываем с каким элементом связана надпись. В `for` мы указываем `id`, связанного элемента. Благодаря этому атрибуту при клике на надпись фокус переходит на нужный элемент.

```
<label for="uname">Name:</label>
<input type="text" id="uname" name="userName" />
```

В этом фрагменте кода надпись связана с текстовым полем, так как в атрибуте `for` значение `uname`. Это `id` текстового поля для ввода имени. У вас может возникнуть вопрос: зачем мы задаём `id` и `name` для элемента управления? Отвечаем:

- `id` — это уникальный идентификатор на странице(значения `id` не должны повторяться). Он используется для обращения к элементу из клиентской части приложения (например, из кода на JavaScript);
- `name` нужно указывать для передачи данных на сервер. Если не указать `name`, в серверной части приложения невозможно будет обратиться к значению элемента. К `name` не предъявляются требования по уникальности. В разных формах на одной и той же странице могут быть элементы с одинаковыми именами;
- Значения для `id` и `name` могут быть одинаковыми.

Второй элемент — многострочное текстовое поле. Для его создания мы использовали тег `textarea`.

```
<textarea id="message" name="userMessage"></textarea>
```

Никаких специфических атрибутов для этого элемента мы не использовали. В коде страницы мы исполь-

зовали теги `div` для простейшего форматирования. Код проекта находится в папке *Contact form*.

Добавим в наш код с регистрационной формой немного оформления с помощью CSS.

Рисунок 10

Новый код страницы:

```
<!DOCTYPE html>

<html lang="en">
  <head>
    <meta charset="utf-8" />
    <style>
      ul {
        list-style: none;
        padding: 0;
        margin: 0;
      }
      label {
        display: inline-block;
        width: 70px;
      }
    </style>
  </head>
  <body>
    <div>
      <h2>We need your opinion!</h2>
      <div>
        Nick: <input type="text">
        Email: <input type="text">
        Phone: <input type="text">
        Message: <input type="text">
        <input type="button" value="Send my message"/>
      </div>
    </div>
  </body>
</html>
```

```

        text-align: right;
    }
    form li + li {
        margin-top: 0.5em;
    }
    /*
        Задаем настройки для ширины формы,
        границ, отступов
    */
    form {
        width: 300px;
        padding: 1em;
        border: 1px solid rgba(23, 4, 56, 0.829);
        border-radius: 2em;
    }
    textarea {
        vertical-align: top;
    }
    input,
    textarea {
        /*
            Устанавливаем один и тот же шрифт
            для input и textarea.
            По умолчанию в textarea используется
            monospace
        */
        font: 1em sans-serif;
        width: 200px;
        border: 1px solid #777;
    }
    input:focus,
    textarea:focus {
        /* Делаем подсветку для элемента в фокусе */
        border-color: #000;
    }
    /* центрируем нашу кнопку */
    input[id="submitBtn"] {

```

```

        margin-left: auto;
        margin-right: auto;
        display: block;
    }
</style>
<title>Send message to us</title>
</head>

<body>
    <h3>We need your opinion!</h3>
    <hr />
    <form>
        <ul>
            <li>
                <label for="nick">Nick:</label>
                <input type="text" id="nick" name="userNick" />
            </li>
            <li>
                <label for="email">Email:</label>
                <input type="email" id="email"
                    name="userEmail" />
            </li>
            <li>
                <label for="phone">Phone:</label>
                <input type="text" id="phone"
                    name="userPhone" />
            </li>
            <li>
                <label for="message">Message:</label>
                <textarea id="message"
                    name="userMessage"></textarea>
            </li>
            <li>
                <input id="submitBtn" type="submit"
                    value="Send my message" />
            </li>
        </ul>
    </form>

```



```
</form>
<hr />
</body>
</html>
```

Изменения не затронули код формы. Мы добавили набор уже знакомых вам стилей для более аккуратного, внешнего вида. Поработайте с кодом формы, чтобы добиться оформления нужного вам. Код проекта находится в папке *Contact form2*.

Продолжим наше знакомство с элементами управления. На очереди два новых элемента: радиокнопка (**radio button, radio**), чекбокс (**checkbox**). Эти элементы знакомы вам. Мы уверены, что вы уже использовали их.

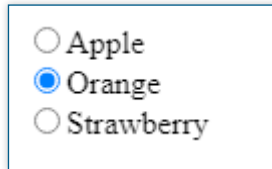


Рисунок 11

Это пример набора радиокнопок. В одной группе радиокнопок обычно имеется возможность выбрать только одну кнопку.

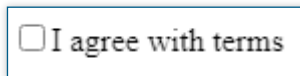


Рисунок 12

Это пример чекбокса. У вас есть возможность поставить или не поставить «галочку».

Создадим новую регистрационную форму с использованием радиокнопок и чекбокса.

Внешний вид нашей страницы:



Please provide information about you

First Name:

Last Name:

Email:

Password:

Subscribe?: ☐

Gender

☐ Male ☐ Female ☐ Other

Рисунок 13

Код страницы:

```
<!DOCTYPE html>

<html lang="en">
  <head>
    <meta charset="utf-8" />
    <style>
      ul {
        list-style: none;
        margin: 0;
        padding: 0;
      }
      li {
        padding: 0.2em;
      }
    </style>
  </head>
  <body>
    <div>
      <h1>Please provide information about you</h1>
      <div>
        <div>First Name:</div><input type="text">
        <div>Last Name:</div><input type="text">
        <div>Email:</div><input type="text">
        <div>Password:</div><input type="password">
        <div>Subscribe?:</div><input type="checkbox">
        <div>Gender</div>
        <div>
          <input type="radio"/> Male
          <input type="radio"/> Female
          <input type="radio"/> Other
        </div>
        <div><input type="button" value="Register"></div>
      </div>
    </div>
  </body>
</html>
```

```

form {
  width: 300px;
  font-family: Verdana, sans-serif;
  font-size: 15px;
  padding: 1em;
  border: 1px solid #000;
  border-radius: 1em;
}
input {
  font: 1em sans-serif;
  width: 150px;
  box-sizing: border-box;
  border: 1px solid #777;
}
input:focus {
  border-color: rgb(25, 11, 100);
}
input[type="checkbox"],
input[type="radio"] {
  width: auto;
}
input[id="submitBtn"] {
  display: block;
  width: 120px;
  margin-left: auto;
  margin-right: auto;
  margin-top: 10px;
}

label span {
  width: 90px;
  text-align: right;
  display: inline-block;
}
</style>
<title>Moonlight news</title>
</head>

```

```

<body>
  <h3>Please provide information about you</h3>
  <hr />
  <form>
    <ul>
      <li>
        <label>
          <span>First Name:</span>
          <input type="text" id="fname"
            name="firstName" />
        </label>
      </li>
      <li>
        <label>
          <span>Last Name:</span>
          <input type="text" id="lName"
            name="lastName" />
        </label>
      </li>
      <li>
        <label>
          <span>Email:</span>
          <input type="email" id="email"
            name="userEmail" />
        </label>
      </li>
      <li>
        <label>
          <span>Password:</span>
          <input type="password" id="password"
            name="userPassword" />
        </label>
      </li>
      <li>
        <label for="news">
          <span>Subscribe?:</span>
          <input type="checkbox" id="news"
            name="newsCheck" />
        </label>
      </li>
    </ul>
  </form>
</body>

```

```

        </label>
    </li>
    <li>
        <p>Gender</p>
        <input type="radio" value="male"
            id="gender1" name="sex" />
        <label for="gender1">Male</label>
        <input type="radio" value="female"
            id="gender2" name="sex" />
        <label for="gender2">Female</label>
        <input type="radio" value="other"
            id="gender3" name="sex" />
        <label for="gender3">Other</label>
    </li>
    <li id="buttons">
        <input id="submitBtn" type="submit"
            value="Register" />
    </li>
</ul>
</form>
<hr />
</body>
</html>

```

В этом примере мы использовали несколько новых приёмов. Наш чекбокс создаётся в этом куске кода:

```

<label for="news">
    <span>Subscribe?:</span>
    <input type="checkbox" id="news"
        name="newsCheck" />
</label>

```

Тут уже есть знакомый нам [label](#) при клике, на который устанавливается галочка в чекбоксе. Это происходит,

потому что мы в атрибуте `for` для `label` указали значение `news` (это `id` нашего чекбокса). Чуть ниже вы узнаете, что мы могли и не указывать `for`. И был бы такой же эффект. Чекбокс был создан при помощи тега `input` у которого `type` равен `checkbox`.

Теперь рассмотрим код по созданию группы радиокнопок.

```
<li>
  <p>
    Gender
  </p>
  <input type="radio" value="male"
        id="gender1" name="sex" />
  <label for="gender1">Male</label>
  <input type="radio" value="female"
        id="gender2" name="sex" />
  <label for="gender2">Female</label>
  <input type="radio" value="other"
        id="gender3" name="sex" />
  <label for="gender3">Other</label>
</li>
```

Для создания радиокнопки используется тег `input` со значением `type` равным `radio`. В нашем коде мы создали три радиокнопки. Для каждой радиокнопки мы указали одинаковое значение для `name`. Мы это делаем для того, чтобы пользователь мог выбрать только одну радиокнопку из группы. Обратите внимание, что мы устанавливаем для каждой из кнопок своё значение в атрибут `value`. Это нужно, чтобы в серверной части вашего приложения можно было определить какая конкретно кнопка была выбрана.

Для создания текстового поля, куда пользователь введёт свой **email** мы использовали новый тег:

```
<input type="email" id="email" name="userEmail" />
```

Этот код создаёт текстовое поле, которое автоматически будет проверять корректность ввода **email**, при попытке отправить данные на сервер.

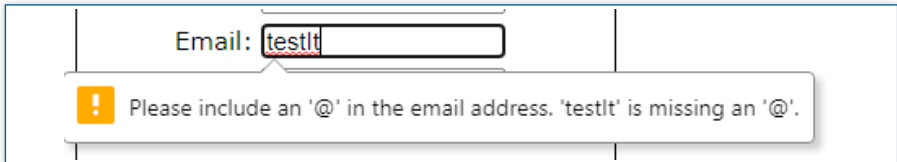


Рисунок 14

Также мы использовали специальный элемент для ввода пароля

```
<input type="password" id="password"
name="userPassword" />
```

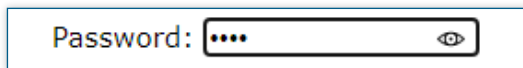


Рисунок 15

Рекомендуем вам использовать эти элементы, а не обычные текстовые поля.

И ещё один важный аспект нашего примера.

```
<label>
  <span>First Name:</span>
  <input type="text" id="fname" name="firstName" />
</label>
```



Рисунок 16

Если кликнуть по надписи **First Name** фокус переходит в текстовое поле рядом с надписью. Этот эффект привычен нам, но мы не указывали атрибут **for** для тега **label**. В этом примере мы применяли альтернативный вариант. Мы разместили текстовое поле и **span** внутри **label**. Такой механизм приводит к тому же результату, что и с **for**

Полный код примера находится в папке *Registration form*.

Продолжаем наше знакомство с элементами управления. На очереди новый элемент — выпадающий список (**combo box**). Внешний вид элемента вам известен.

Список в закрытом состоянии:

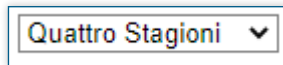


Рисунок 17

Список в открытом состоянии:

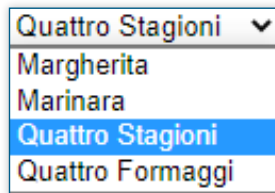


Рисунок 18

Список обычно используется для выбора одного конкретного элемента.

Для знакомства с ним создадим форму заказа пиццы. Внешний вид нашей формы:

Рисунок 19

Исходный код страницы:

```
<html lang="en">
<head>
<style>
  h4 {
    font-family: Verdana, sans-serif;
  }
  form {
    width: 250px;
    font-family: Verdana, sans-serif;
    font-size: 12px;
  }
  input {
    font: 1em Verdana, sans-serif;
    width: 130px;
```

```
        box-sizing: border-box;
        border: 1px solid #777;
    }
    input:focus {
        border-color: rgb(255, 255, 0);
    }
    div {
        padding: 0.2em;
    }
    select {
        width: 130px;
        box-sizing: border-box;
        border: 1px solid #777;
    }
    ul {
        list-style: none;
        padding: 0;
        margin: 0;
    }
    fieldset {
        margin: 0.5em;
        width: 200px;
    }
    label span {
        width: 90px;
        text-align: right;
        display: inline-block;
    }
    input[type="radio"] {
        width: auto;
    }
    button[type="submit"] {
        width: 100px;
        height: 30px;
        font-family: Verdana, sans-serif;
        font-size: 12px;
        margin-left: 80px;
    }

```

```

    }
</style>
<meta charset="utf-8" />
<title>Order pizza</title>
</head>
<body>
  <h4>Fill details for pizza order:</h4>
  <form>
    <div>
      <label>
        <span>Name:</span>
        <input type="text" id="name" name="userName" />
      </label>
    </div>
    <div>
      <label>
        <span>Phone:</span>
        <input type="text" id="phone"
          name="userPhone" />
      </label>
    </div>
    <div>
      <label>
        <span>Address:</span>
        <input type="text" id="address"
          name="userAddress" />
      </label>
    </div>
    <div>
      <label>
        <span>Choose pizza:</span>
        <select id="nameOfPizza" name="nameOfPizza">
          <option>Margherita</option>
          <option>Marinara</option>
          <option>Quattro Stagioni</option>
          <option>Quattro Formaggi</option>
        </select>
      </label>
    </div>
  </form>
</body>
</html>

```

```

</div>
<fieldset>
  <legend>Select size:</legend>
  <ul>
    <li>
      <label>
        <span>Small</span>
        <input type="radio" id="small"
              name="size" value="small" />
      </label>
    </li>
    <li>
      <label>
        <span>Medium</span>
        <input
          type="radio"
          id="medium"
          name="size"
          value="medium"
          checked
        />
      </label>
    </li>
    <li>
      <label>
        <span>Large </span>
        <input type="radio" id="medium"
              name="size" value="medium" />
      </label>
    </li>
  </ul>
</fieldset>
<div>
  <button type="submit">Order</button>
</div>
</form>
</body>
</html>

```

Для создания списка используется тег `<select>`. Элементы списка описываются внутри него. Отдельный элемент списка создаётся с помощью тега `<option>`. Код по созданию списка:

```
<select id="nameOfPizza" name="nameOfPizza">
  <option>Margherita</option>
  <option>Marinara</option>
  <option>Quattro Stagioni</option>
  <option>Quattro Formaggi</option>
</select>
```

Количество `<option>` равно количеству вариантов пицц. В коде страницы мы использовали ещё один новый тег — `<fieldset>`. Он предназначен для группировки элементов формы. Результат его работы выглядит так:

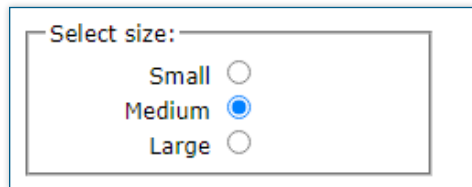


Рисунок 20

Браузер отображает рамку вокруг элементов, сгруппированных с помощью `<fieldset>`. Код элементов содержится внутри `<fieldset>`.

```
<fieldset>
  теги
</fieldset>
```

Для заголовка `fieldset` используется тег `legend`.

```
<fieldset>
  <legend>Select size:</legend>
```

Также обратите внимание, что мы выбрали одну из радиокнопок (размер пиццы **Medium**). Мы использовали атрибут **checked**, чтобы добиться такого результата.

```
<input
  type="radio"
  id="medium"
  name="size"
  value="medium"
  checked
/>
```


Код проекта в папке *Pizza form*.

И ещё один пример. Мы создадим форму для регистрации на кинопортале. Цель этой страницы показать элементы форм, которые мы ранее не использовали.

Внешний вид страницы:

Fill information about you and your cinema experience:

Name:

Birthday: 

Email:

Blog:

Genre:

How you rate yourself:

Рисунок 21

На этой странице мы использовали следующие новые возможности: поле для выбора даты, поле для ввода URL, текстовое поле с автозавершением, текстовое поле с контролем значения ([spin control](#)).

Начнем с кода страницы. Мы приведем лишь код формы. Полный код страницы можно будет найти в папке *Cinema form*.

```
<form>
  <div>
    <label>
      <span>Name:</span>
      <input type="text" id="name" name="userName" />
    </label>
  </div>
  <div>
    <label>
      <span>Birthday:</span>
      <input type="date" name="birthday"
        id="userBirthday">
    </label>
  </div>
  <div>
    <label>
      <span>Email:</span>
      <input type="email" id="email"
        name="userEmail" />
    </label>
  </div>
  <div>
    <label>
      <span>Blog:</span>
      <input type="url" id="url" name="userUrl">
    </label>
  </div>
</div>
```

```

<label>
  <span>Genre:</span>
  <input type="text"
    name="genre"
    id="userGenre"
    list="genreSuggestions">
  <datalist id="genreSuggestions">
    <option>Comedy</option>
    <option>Horror</option>
    <option>Drama</option>
    <option>Thriller</option>
    <option>Science Fiction</option>
    <option>Fantasy</option>
  </datalist>
</label>
</div>
<div>
  <label>
    <span>How you rate yourself:</span>
    <input type="number" name="userRating"
      id="rating" min="1" max="12"
      step="1">
  </label>
</div>
<div>
  <button type="submit">Send my data</button>
</div>
</form>

```

Разбираем по шагам. Начнем с элемента для выбора даты. Ввод даты всегда был головной болью для HTML-разработчиков. С появлением HTML5 эта боль стала значительно меньше, потому что появился набор элементов управления, которые решают эту задачу.

```
<input type="date" name="birthday" id="userBirthday">
```


Для создания элемента управления с датой мы используем `input` с `type = date`.

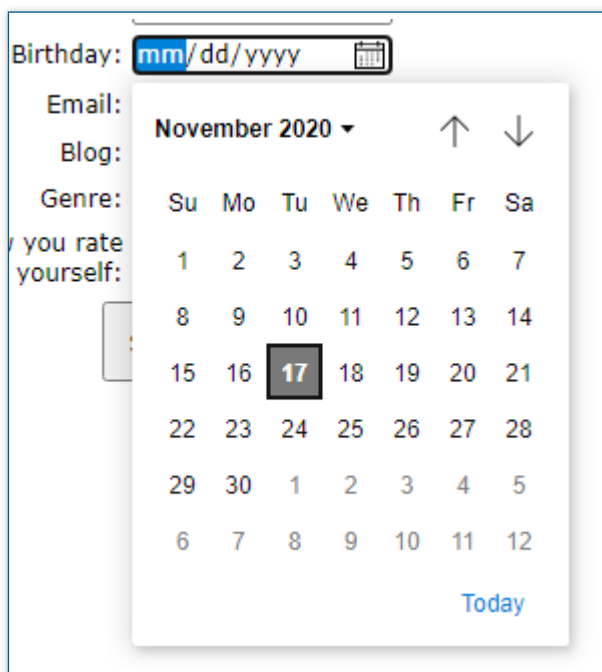


Рисунок 22

К сожалению, пока не все браузеры поддерживают этот элемент. Обязательно проверяете поддержку того или иного тега в браузере. Например, если вам необходимо проверить поддержку `date`, вы можете это сделать по этой [ссылке](#).

Второй новый элемент — это поле для ввода `URL`. Оно проверяет корректность адреса. Обратите внимание, этот элемент не проверяет существует ли в реальности ваш `URL`. Его цель только проверка значения на формат данных. Например:

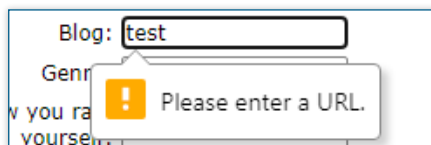


Рисунок 23

Для создания такого поля мы использовали следующий код:

```
<input type="url" id="url" name="userUrl">
```

Третий новый элемент — текстовое поле с автозавершением. Внешний вид до ввода текста:

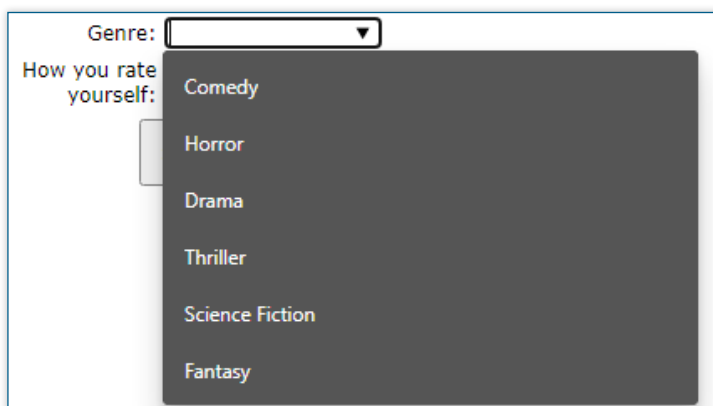


Рисунок 24

Вводим текст:



Рисунок 25

Код для создания элемента:

```
<input type="text" name="genre" id="userGenre"
      list="genreSuggestions" />
<datalist id="genreSuggestions">
  <option>Comedy</option>
  <option>Horror</option>
  <option>Drama</option>
  <option>Thriller</option>
  <option>Science Fiction</option>
  <option>Fantasy</option>
</datalist>
```

В текстовом поле мы добавили один атрибут **list**. Он используется для указания списка, откуда будут браться значения для автозавершения. В **list** записано **id** списка.

Сам список определен ниже с помощью тега **datalist**. Тег **option** создаёт конкретный элемент списка.

И последний новый элемент — счётчик (иногда его называют **spin control**). Внешний вид:

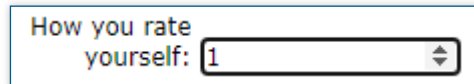


Рисунок 26

Код для создания:

```
<input type="number"
      name="userRating"
      id="rating"
      min="1"
      max="12"
      step="1"
      >
```

Счётчик создаётся с помощью атрибута `type = number`. Мы указали, что минимально возможное значение 1, максимальное значение 12. При нажатии на стрелки справа, значение будет изменяться на один.

Код проекта в папке *Cinema form*.

Мы рассказали вам о работе с формами. Обязательно поэкспериментируйте с примерами и закрепите новый материал выполнив домашнее задание.

Домашнее задание

1. Создайте страницу для заказа суши. Для заказа нужно указать такую информацию:
 - Имя;
 - Адрес;
 - Телефон;
 - Email;
 - Количество палочек;
 - Вид палочек;
 - Обычные;
 - Учебные;
 - Список для выбора суши;
 - Нужно ли класть салфетки?
 - Дополнительный комментарий.
2. Создайте страницу для регистрации в онлайн-игре. Для регистрации нужно указать такую информацию:
 - Ник;
 - Email;
 - Дата рождения;
 - Пол;
 - Страна (список с автозавершением);
 - Тип героя;
 - Игровой опыт в годах (от 0 до 60).