

# HTML tables, lists, forms, images, audio, video

Vira Huskova

HTML:

tables

lists

forms

images

audio, video

div

SVG and Canvas overview

# HTML-таблицы

1	<b>H</b>	Водород	
			s <sup>1</sup> 1.00794
3	<b>Li</b>	Литий	
			s <sup>1</sup> 6.941 s <sup>2</sup> 9.0121
11	<b>Na</b>	Натрий	
			s <sup>1</sup> 22.9897 s <sup>2</sup> 24.3050
19	<b>K</b>	Калий	
			s <sup>1</sup> 39.0983 s <sup>2</sup> 40.078
37	<b>Rb</b>	Рубидий	
			s <sup>1</sup> 85.468 s <sup>2</sup> 87.62
55	<b>Cs</b>	Цезий	
			s <sup>1</sup> 132.91 s <sup>2</sup> 137.33
87	<b>Fr</b>	Франций	
			s <sup>1</sup> 223.02 s <sup>2</sup> 226.02
119	<b>Uue</b>	Унуненний	
			s <sup>1</sup> 316 s <sup>2</sup> 320

2	<b>He</b>	Гелий	
			s <sup>2</sup> 4.0026
5	<b>B</b>	Бор	
			s <sup>2</sup> p <sup>1</sup> 10.811
6	<b>C</b>	Углерод	
			s <sup>2</sup> p <sup>2</sup> 12.0107
7	<b>N</b>	Азот	
			s <sup>2</sup> p <sup>3</sup> 14.0067
8	<b>O</b>	Кислород	
			s <sup>2</sup> p <sup>4</sup> 15.9994
9	<b>F</b>	Фтор	
			s <sup>2</sup> p <sup>5</sup> 18.9984
10	<b>Ne</b>	Неон	
			s <sup>2</sup> p <sup>6</sup> 20.1797
13	<b>Al</b>	Алюминий	
			s <sup>2</sup> p <sup>1</sup> 26.9815
14	<b>Si</b>	Кремний	
			s <sup>2</sup> p <sup>2</sup> 28.0855
15	<b>P</b>	Фосфор	
			s <sup>2</sup> p <sup>3</sup> 30.9737
16	<b>S</b>	Сера	
			s <sup>2</sup> p <sup>4</sup> 32.065
17	<b>Cl</b>	Хлор	
			s <sup>2</sup> p <sup>5</sup> 35.453
18	<b>Ar</b>	Аргон	
			s <sup>2</sup> p <sup>6</sup> 39.948
31	<b>Ga</b>	Галлий	
			s <sup>2</sup> p <sup>1</sup> 69.723
32	<b>Ge</b>	Германий	
			s <sup>2</sup> p <sup>2</sup> 72.63
33	<b>As</b>	Мышьяк	
			s <sup>2</sup> p <sup>3</sup> 74.9216
34	<b>Se</b>	Селен	
			s <sup>2</sup> p <sup>4</sup> 78.96
35	<b>Br</b>	Бром	
			s <sup>2</sup> p <sup>5</sup> 79.904
36	<b>Kr</b>	Криптон	
			s <sup>2</sup> p <sup>6</sup> 83.798
31	<b>In</b>	Индий	
			s <sup>2</sup> p <sup>1</sup> 114.82
50	<b>Sn</b>	Олово	
			s <sup>2</sup> p <sup>2</sup> 118.71
51	<b>Sb</b>	Сурьма	
			s <sup>2</sup> p <sup>3</sup> 121.76
52	<b>Te</b>	Теллур	
			s <sup>2</sup> p <sup>4</sup> 127.60
53	<b>I</b>	Иод	
			s <sup>2</sup> p <sup>5</sup> 126.90
54	<b>Xe</b>	Ксенон	
			s <sup>2</sup> p <sup>6</sup> 131.29
72	<b>Hf</b>	Гафний	
			s <sup>2</sup> d <sup>2</sup> 178.49
73	<b>Ta</b>	Тантал	
			s <sup>2</sup> d <sup>3</sup> 180.94
74	<b>W</b>	Вольфрам	
			s <sup>2</sup> d <sup>4</sup> 183.85
75	<b>Re</b>	Рений	
			s <sup>2</sup> d <sup>5</sup> 186.2
76	<b>Os</b>	Оsmий	
			s <sup>2</sup> d <sup>7</sup> 190.2
77	<b>Ir</b>	Иридий	
			s <sup>2</sup> d <sup>7</sup> 192.2
78	<b>Pt</b>	Платина	
			s <sup>1</sup> d <sup>9</sup> 195.09
79	<b>Au</b>	Золото	
			s <sup>1</sup> d <sup>10</sup> 196.96
80	<b>Hg</b>	Ртуть	
			s <sup>2</sup> d <sup>10</sup> 200.59
81	<b>Tl</b>	Таллий	
			s <sup>2</sup> p <sup>1</sup> 204.37
82	<b>Pb</b>	Свинец	
			s <sup>2</sup> p <sup>2</sup> 207.19
83	<b>Bi</b>	Висмут	
			s <sup>2</sup> p <sup>3</sup> 208.98
84	<b>Po</b>	Полоний	
			s <sup>2</sup> p <sup>4</sup> 208.98
85	<b>At</b>	Астат	
			s <sup>2</sup> p <sup>5</sup> 209.98
86	<b>Rn</b>	Радон	
			s <sup>2</sup> p <sup>6</sup> 222.01
104	<b>Rf</b>	Резерфордий	
			s <sup>2</sup> d <sup>14</sup> 261
105	<b>Db</b>	Дубний	
			s <sup>2</sup> d <sup>3</sup> 268
106	<b>Sg</b>	Сиборгий	
			s <sup>2</sup> d <sup>4</sup> 271
107	<b>Bh</b>	Борий	
			s <sup>2</sup> d <sup>5</sup> 267
108	<b>Hs</b>	Хассий	
			s <sup>2</sup> d <sup>7</sup> 269
109	<b>Mt</b>	Мейнитерий	
			s <sup>1</sup> d <sup>9</sup> 276
110	<b>Ds</b>	Дармштадтий	
			s <sup>1</sup> d <sup>9</sup> 281
111	<b>Rg</b>	Рентгений	
			s <sup>1</sup> d <sup>10</sup> 280
112	<b>Cn</b>	Коперниций	
			s <sup>2</sup> d <sup>10</sup> 285
113	<b>Uut</b>	Унунтрий	
			s <sup>2</sup> p <sup>1</sup> 284
114	<b>Uuo</b>	Унунквадий	
			s <sup>2</sup> p <sup>2</sup> 289
115	<b>Uup</b>	Унунпентий	
			s <sup>2</sup> p <sup>3</sup> 288
116	<b>Uuh</b>	Унунгексий	
			s <sup>2</sup> p <sup>4</sup> 293
117	<b>Uus</b>	Унунсептий	
			s <sup>2</sup> p <sup>5</sup> 294
118	<b>Uuo</b>	Унуноктий	
			s <sup>2</sup> p <sup>6</sup> 294

57	<b>La</b>	Лантан	
			s <sup>2</sup> d <sup>1</sup> 138.91
58	<b>Ce</b>	Церий	
			a <sup>2</sup> 140.12
59	<b>Pr</b>	Празеодим	
			s <sup>2</sup> p <sup>3</sup> 140.90
60	<b>Nd</b>	Неодим	
			s <sup>2</sup> d <sup>4</sup> 144.24
61	<b>Pm</b>	Прометий	
			s <sup>2</sup> f <sup>5</sup> 145
62	<b>Sm</b>	Самарий	
			s <sup>2</sup> f <sup>6</sup> 150.35
63	<b>Eu</b>	Европий	
			s <sup>2</sup> f <sup>7</sup> 151.96
64	<b>Gd</b>	Гадолиний	
			s <sup>2</sup> d <sup>7</sup> f <sup>8</sup> 157.25
65	<b>Tb</b>	Тербий	
			s <sup>2</sup> d <sup>10</sup> 158.92
66	<b>Dy</b>	Диспрозий	
			s <sup>2</sup> d <sup>10</sup> 162.50
67	<b>Ho</b>	Гольмий	
			s <sup>2</sup> f <sup>11</sup> 164.93
68	<b>Er</b>	Эрбий	
			s <sup>2</sup> f <sup>12</sup> 167.26
69	<b>Tm</b>	Тулий	
			s <sup>2</sup> f <sup>13</sup> 168.93
70	<b>Yb</b>	Иттербий	
			s <sup>2</sup> f <sup>14</sup> 173.04
71	<b>Lu</b>	Лютесций	
			s <sup>2</sup> d <sup>14</sup> 174.97
89	<b>Ac</b>	Актиний	
			s <sup>2</sup> d <sup>1</sup> 227.02
90	<b>Th</b>	Торий	
			s <sup>2</sup> d <sup>2</sup> 232.03
91	<b>Pa</b>	Протактиний	
			s <sup>2</sup> d <sup>1</sup> f <sup>2</sup> 231.03
92	<b>U</b>	Уран	
			s <sup>2</sup> d <sup>1</sup> f <sup>3</sup> 238.02
93	<b>Np</b>	Нептуний	
			s <sup>2</sup> d <sup>1</sup> f <sup>4</sup> 237.04
94	<b>Pu</b>	Плутоний	
			s <sup>2</sup> f <sup>6</sup> 244.06
95	<b>Am</b>	Америций	
			s <sup>2</sup> f <sup>7</sup> 243.06
96	<b>Cm</b>	Кюрий	
			s <sup>2</sup> f <sup>8</sup> 247.07
97	<b>Bk</b>	Берклий	
			s <sup>2</sup> f <sup>9</sup> 247.07
98	<b>Cf</b>	Калифорний	
			s <sup>2</sup> f <sup>10</sup> 251.07
99	<b>Es</b>	Эйнштейний	
			s <sup>2</sup> f <sup>11</sup> 252.08
100	<b>Fm</b>	Фермий	
			s <sup>2</sup> f <sup>12</sup> 257.08
101	<b>Md</b>	Менделевий	
			s <sup>2</sup> f <sup>13</sup> 258.09
102	<b>No</b>	Нобелний	
			s <sup>2</sup> f <sup>14</sup> 259.10
103	<b>Lr</b>	Лоуренсий	
			s <sup>2</sup> d <sup>14</sup> 260.10

**HTML-таблицы** упорядочивают и выводят на экран данные с помощью **строк или столбцов**.

Таблицы состоят из ячеек, образующихся при пересечении строк и столбцов.

**Ячейки таблиц** могут содержать любые HTML-элементы, такие как заголовки, списки, текст, изображения, элементы форм, а также другие таблицы.

Ячейка		Ячейка
Ячейка	Ячейка	Ячейка
Ячейка	Ячейка	Ячейка

Каждой таблице можно добавить связанный с ней заголовок, расположив его перед таблицей или после неё.

## Как создать таблицу

Таблица создаётся при помощи парного тега `<table></table>`.

Данный тег является контейнером для элементов таблицы и все элементы должны находиться **внутри него**. Например, с помощью данной разметки можно создать таблицу, состоящую из **двух столбцов и двух строк**:

```
<table>
<tr><th>текст заголовка</th><th>текст заголовка</th></tr> <!--ячейками заголовков-->
<tr><td>данные</td><td>данные</td></tr> <!--ряд с ячейками тела таблицы-->
</table>
```

текст заголовка	текст заголовка
данные	данные

По умолчанию таблица и ячейки не имеют видимых границ.  
Границы задаются с помощью свойства `border`:

Промежутки между ячейками таблицы убираются с помощью свойства `table {border-collapse: collapse;}`.

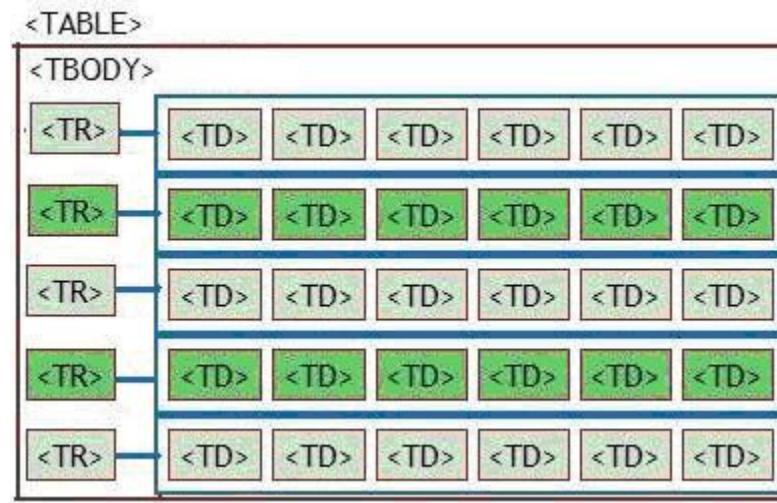
Ширина таблицы по умолчанию равна ширине её внутреннего содержимого. Чтобы установить ширину, нужно задать значение для свойства `width`.

Если для ячеек таблицы заданы внутренние отступы и границы, то ширина таблицы будет включать в себя следующие значения: `padding-left` и `padding-right`, ширина `border-left` плюс ширина `border-right` последней ячейки в ряду.

Если заданы ширина и границы ячеек, то ширина таблицы будет складываться из ширины ячеек плюс ширина `border-left` и ширина `border-right` последней ячейки в ряду.

## Строки (ряды) таблицы

Строки или ряды таблицы создаются с помощью тега `<tr>`. Количество горизонтальных строк таблицы определяется количеством парных тегов `<tr></tr>`.



## Ячейка заголовка столбца таблицы

Элемент `<th>` создаёт заголовок столбца – специальную ячейку, текст в которой выделяется полужирным. Количество ячеек заголовка определяется количеством пар тегов `<th></th>`. Для элемента доступны атрибуты `colspan`, `rowspan`, `headers`.

```
<table>
<tr><th>ячейка заголовка</th><th>ячейка заголовка</th></tr>
</table>
```

## Ячейка тела таблицы

Элемент `<td>` создаёт ячейки таблицы, внутрь которых помещаются данные таблицы. Парные теги `<td></td>`, расположенные в одном ряду, определяют количество ячеек в строке таблицы. Количество пар ячеек `<td>` должно быть равно количеству пар ячеек `<th>`. Для элемента доступны атрибуты `colspan`, `rowspan`, `headers`.

```
<table>
<tr><th>ячейка заголовка</th><th>ячейка заголовка</th></tr>
<tr><td>ячейка тела таблицы</td><td>ячейка тела таблицы</td></tr>
</table>
```

## Подпись (заголовок) к таблице

Элемент `<caption>` создает подпись таблицы. Добавляется непосредственно после тега `<table>`, вне строки или ячейки.

```
<table>
<caption>Перечень
продуктов</caption>
<tr>
<th>...</th>
</tr>
<tr>...
</tr>
</table>
```

### Перечень продуктов

**№ п/п Наименование товара Ед. изм. Количество Цена за ед. изм., usd. Стоимость, usd.**

1.	Томаты свежие	кг	15,20	69,00	1048,80
2.	Огурцы свежие	кг	2,50	48,00	120,00

ИТОГО: 1168,80

## Группирование строк и столбцов таблицы

Элемент `<colgroup>` создает структурную группу столбцов, выделяя логически однородные ячейки. Группирует один или более столбцов для единого форматирования, позволяя применить стили к столбцам вместо того, чтобы повторять стили для каждой ячейки и для каждой строки. Добавляется непосредственно после тегов `<table>` и `<caption>`.

Элемент `<col>` формирует группы столбцов, которые делят таблицу на разделы, не относящиеся к общей структуре, т.е. не содержащие информацию одного типа. Позволяет задавать свойства столбцов для каждого столбца в пределах элемента `<colgroup>`. С помощью атрибута `style` можно изменить основной цвет фона ячеек. Для элемента `<col>` доступен атрибут `span`, задающий количество столбцов для объединения.

Название	Цвет	Твердость по Моосу	Формула
Алмаз	Белый	10	C
Рубин	Красный	9	$\text{Al}_2\text{O}_3$
Аметист	Голубой	7	$\text{SiO}_2$
Изумруд	Зеленый	8	$\text{Be}_3\text{Al}_2(\text{SiO}_3)_6$
Сапфир	Голубой	9	$\text{Al}_2\text{O}_3$

## Группировка разделов таблицы

Элемент `<thead>` создает группу заголовков для строк таблицы с целью задания единого оформления. Используется в сочетании с элементами `<tbody>` и `<tfoot>` для указания каждой части таблицы.

Элемент должен быть использован в следующем порядке: как дочерний элемент `<table>`, после `<caption>` и `<colgroup>`, и перед `<tbody>`, `<tfoot>` и `<tr>` элементами. В пределах одной таблицы можно использовать один раз.

Элемент `<tbody>` группирует основное содержимое таблицы. Используется в сочетании с элементами `<thead>` и `<tfoot>`.

Элемент `<tfoot>` создает группу строк для представления информации о суммах или итогах, расположенную в нижней части таблицы. Используется в таблице один раз. Располагается после тега `<thead>`, перед тегами `<tbody>` и `<tr>`.

## Объединение ячеек таблицы

Атрибуты `colspan` и `rowspan` объединяют ячейки таблицы. Атрибут `colspan` задает количество ячеек, объединенных по горизонтали, а `rowspan` — по вертикали.

Браузер	Посещения	
	Количество	В процентах
Mozilla Firefox	163	59%
Google Chrome	78	28%
Safari	35	13%

# Атрибуты элементов таблицы

Атрибут	Описание, принимаемое значение
<code>colspan</code>	<p>Количество ячеек в строке для объединения по горизонтали.</p> <pre>&lt;td colspan="3"&gt;</pre> <p>Возможные значения: число от 1 до 999.</p>
<code>headers</code>	<p>Задает список ячеек заголовка, содержащих информацию о заголовке текущей ячейки данных. Предназначен для речевых браузеров.</p> <pre>&lt;th id="идентификатор"&gt;...&lt;/th&gt; &lt;th headers="идентификатор"&gt;...&lt;/th&gt;</pre> <p>Принимаемые значения: список имен ячеек, разделенных пробелами; эти имена должны быть присвоены ячейкам через их атрибут <code>id</code>.</p>
<code>rowspan</code>	<p>Количество ячеек в столбце для объединения по вертикали.</p> <pre>&lt;td rowspan="2"&gt;</pre> <p>Возможные значения: число от 1 до 999.</p>
<code>span</code>	<p>Количество колонок, объединяемых для задания единого стиля, по умолчанию равно 1.</p> <pre>&lt;col span="2"&gt;</pre> <p>Принимаемые значения: любое целое положительное число.</p>

# Списки

HTML-списки используются для группировки связанных между собой фрагментов информации.

Существует три вида списков:

**маркированный список** – `<ul>` – каждый элемент списка `<li>` отмечается маркером,

**нумерованный список** – `<ol>` – каждый элемент списка `<li>` отмечается цифрой,

**список определений** – `<dl>` – состоит из пар термин `<dt>` – `<dd>` определение.

1 Первый пункт списка

1 Подпункт

2 Подпункт

3 Подпункт

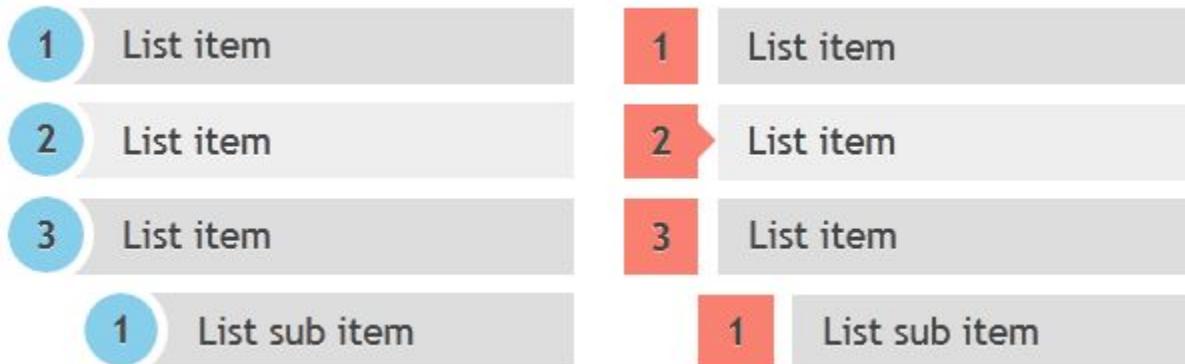
2 Второй

3 Третий

Каждый список представляет собой контейнер, внутри которого располагаются элементы списка или пары термин-определение.

Элементы списка ведут себя как блочные элементы, располагаясь друг под другом и занимая всю ширину блока-контейнера.

Каждый элемент списка имеет дополнительный блок, расположенный сбоку, который не участвует в компоновке.



## Маркированный список

Маркированный список представляет собой неупорядоченный список (*от англ. Unordered List*). Создаётся с помощью парного тега `<ul></ul>`. В качестве маркера элемента списка выступает метка, например, закрашенный кружок.

Каждый элемент списка создаётся с помощью парного тега `<li></li>` (*от англ. List Item*).

Для тега `<ul>` доступны глобальные атрибуты.

```
<ul>
    <li>Microsoft</li>
    <li>Google</li>
    <li>Apple</li>
    <li>IBM</li>
</ul>
```

Тип списка	Код HTML	Пример
Список с маркерами в виде круга	<pre>&lt;ul type="disc"&gt; &lt;li&gt;...&lt;/li&gt; &lt;/ul&gt;</pre>	<ul style="list-style-type: none"> <li>• Первый</li> <li>• Второй</li> <li>• Третий</li> </ul>
Список с маркерами в виде окружности	<pre>&lt;ul type="circle"&gt; &lt;li&gt;...&lt;/li&gt; &lt;/ul&gt;</pre>	<ul style="list-style-type: none"> <li>◦ Первый</li> <li>◦ Второй</li> <li>◦ Третий</li> </ul>
Список с квадратными маркерами	<pre>&lt;ul type="square"&gt; &lt;li&gt;...&lt;/li&gt; &lt;/ul&gt;</pre>	<ul style="list-style-type: none"> <li>▪ Первый</li> <li>▪ Второй</li> <li>▪ Третий</li> </ul>

## Нумерованный список

Нумерованный список создаётся с помощью парного тега `<ol></ol>`. Каждый пункт списка также создаётся с помощью элемента `<li>`. Браузер нумерует элементы по порядку автоматически и если удалить один или несколько элементов такого списка, то остальные номера будут автоматически пересчитаны.

Для тега `<li>` доступен атрибут `value`, который позволяет изменить номер по умолчанию для выбранного элемента списка. Например, если для первого пункта списка задать `<li value="10">`, то остальная нумерация будет пересчитана относительно нового значения.

Тип списка	Код HTML	Пример
Арабские числа	<ol type="1"> <li>...</li> </ol>	1. Чебурашка 2. Крокодил Гена 3. Шапокляк
Прописные буквы латинского алфавита	<ol type="A"> <li>...</li> </ol>	A. Чебурашка B. Крокодил Гена C. Шапокляк
Строчные буквы латинского алфавита	<ol type="a"> <li>...</li> </ol>	a. Чебурашка b. Крокодил Гена c. Шапокляк
Римские числа в верхнем регистре	<ol type="I"> <li>...</li> </ol>	I. Чебурашка II. Крокодил Гена III. Шапокляк
Римские числа в нижнем регистре	<ol type="i"> <li>...</li> </ol>	i. Чебурашка ii. Крокодил Гена iii. Шапокляк

Для тега `<ol>` доступны следующие атрибуты:

Атрибут	Описание, принимаемое значение
<code>reversed</code>	Атрибут <code>reversed</code> задает отображение списка в обратном порядке (например, 9, 8, 7...).
<code>start</code>	Атрибут <code>start</code> задает начальное значение, от которого пойдет отсчет нумерации, например, конструкция <code>&lt;ol start="10"&gt;</code> первому пункту присвоит порядковый номер «10». Также можно одновременно задавать тип нумерации, например, <code>&lt;ol type="I" start="10"&gt;</code> .
<code>type</code>	Атрибут <code>type</code> задает вид маркера для использования в списке (в виде букв или цифр). Принимаемые значения: <code>1</code> — значение по умолчанию, десятичная нумерация. <code>A</code> — нумерация списка в алфавитном порядке, заглавные буквы (A, B, C, D). <code>a</code> — нумерация списка в алфавитном порядке, строчные буквы (a, b, c, d). <code>I</code> — нумерация римскими заглавными цифрами (I, II, III, IV). <code>i</code> — нумерация римскими строчными цифрами (i, ii, iii, iv).

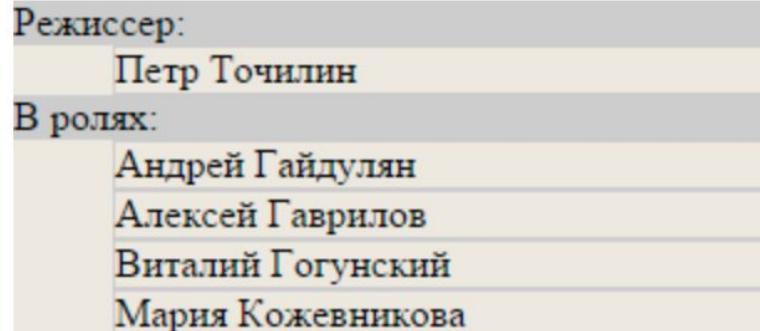
```
<ol>
  <li>Microsoft</li>
  <li>Google</li>
  <li>Apple</li>
  <li>IBM</li>
</ol>
```

## Список определений

Списки определений создаются с помощью тега `<dl></dl>`. Для добавления термина применяется тег `<dt></dt>`, а для вставки определения – тег `<dd></dd>`.

Для тегов `<dl>`, `<dt>` и `<dd>` доступны универсальные атрибуты (см. предыдущую презентацию).

```
<dl>
  <dt>Режиссер:</dt>
    <dd>Петр Точилин</dd>
  <dt>В ролях:</dt>
    <dd>Андрей Гайдулян</dd>
    <dd>Алексей Гаврилов</dd>
    <dd>Виталий Гогунский</dd>
    <dd>Мария Кожевникова</dd>
</dl>
```



## Вложенный список

Возможностей простых списков не хватает, например, при создании оглавления никак не обойтись без вложенных пунктов. Разметка для вложенного списка будет следующей

```
<ul>
<li>Пункт 1.</li>
<li>Пункт 2.
    <ul>
        <li>Подпункт 2.1.</li>
        <li>Подпункт 2.2.
            <ul>
                <li>Подпункт 2.2.1.</li>
                <li>Подпункт 2.2.2.</li>
            </ul>
        </li>
        <li>Подпункт 2.3.</li>
    </ul>
</li>
<li>Пункт 3.</li>
</ul>
```

- Пункт 1.
- Пункт 2.
  - Подпункт 2.1.
  - Подпункт 2.2.
    - Подпункт 2.2.1.
    - Подпункт 2.2.2.
  - Подпункт 2.3.
- Пункт 3.

Пример: Выпадающее меню

# Многоуровневый нумерованный список

Многоуровневый список используется для отображения элементов списка на разных уровнях с различными отступами. Разметка для многоуровневого нумерованного списка будет следующей:

```
<ol>
  <li>пункт</li> <!--1.-->
  <li>пункт
    <ol>
      <li>пункт</li> <!--2.1.-->
      <li>пункт</li> <!--2.2.-->
      <li>пункт
        <ol>
          <li>пункт</li> <!--2.3.1.-->
          <li>пункт</li> <!--2.3.2.-->
          <li>пункт</li> <!--2.3.3.-->
        </ol>
      </li> <!--2.3.-->
      <li>пункт</li> <!--2.4.-->
    </ol>
  </li> <!--2.-->
  <li>пункт</li> <!--3.-->
  <li>пункт</li> <!--4.-->
</ol>
```

Такая разметка по умолчанию создаст для каждого вложенного списка новую нумерацию, начинающуюся с единицы.

Чтобы сделать вложенную нумерацию, нужно использовать следующие свойства:

`counter-reset` сбрасывает один или несколько счётчиков, задавая значение для сброса;

`counter-increment` задаёт значение приращения счётчика, т.е. с каким шагом будет нумероваться каждый последующий пункт;

`content` – генерируемое содержимое, в данном случае отвечает за вывод номера перед каждым пунктом списка.

## Counter-reset

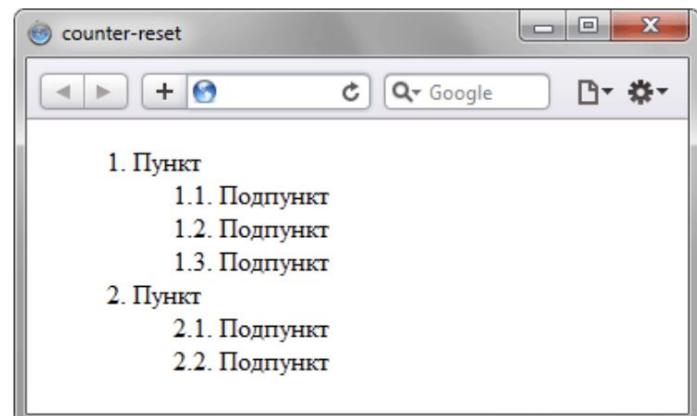
Устанавливает идентификатор, в котором будет храниться счётчик отображений определенного элемента, а также начальное значение счётика. Такой счётик может выводиться с помощью свойства content и псевдоэлементов :after и :before.

**none** Запрещает инициацию счётика для текущего селектора.

**inherit** Наследует значение родителя.

**идентификатор** Задаёт одну или несколько переменных, в которых будет храниться значение счётика. Значения разделяются между собой пробелом.

**целое число** Начальное значение каждого идентификатора. По умолчанию равно 0.



## counter-increment

Стилевое свойство counter-increment предназначено для увеличения значения счётчика приращений, который задается свойством counter-reset.

Такой счётчик подсчитывает количество отображений элементов на странице и может выводиться с помощью свойства content и псевдоэлементов :after и :before. Это позволяет создавать списки (в том числе многоуровневые), в которых нумерация и вид задаются через стили.

```
counter-increment: none | inherit | идентификатор | целое число
```

**none** Запрещает увеличение счетчика для текущего селектора.

**inherit** Наследует значение родителя.

**идентификатор** Задает одну или несколько переменных, для которых требуется изменить значение счетчика. Переменные разделяются между собой пробелом.

**целое число** Определяет значение приращения счетчика. По умолчанию оно равно 1. Допускается использовать отрицательные и нулевые значения.

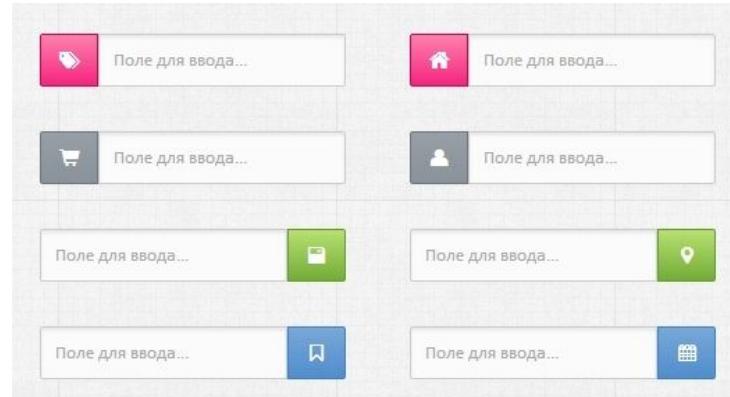
Код	Результат
<pre>LI { list-style-type: none; } OL { counter-reset: list -1; } LI:before {     counter-increment: list;     content: counter(list) ". "; }</pre>	<p>Список начинается с нуля.</p> <p>0, 1, 2</p>
<pre>LI { list-style-type: none; } OL { counter-reset: list; } LI:before {     counter-increment: list 2;     content: counter(list) ". "; }</pre>	<p>Выводятся все четные числа.</p> <p>2, 4, 6</p>
<pre>LI { list-style-type: none; } OL { counter-reset: list -1; } LI:before {     counter-increment: list list;     content: counter(list) ". "; }</pre>	<p>Выводятся все нечетные числа.</p> <p>1, 3, 5</p>
<pre>LI { list-style-type: none; } OL { counter-reset: list 9; } LI:before {     counter-increment: list;     content: counter(list) ". "; }</pre>	<p>Список начинается с 10.</p> <p>10, 11, 12</p>

## HTML5-формы

HTML-формы являются элементами управления, которые применяются для сбора информации от посетителей веб-сайта.

Веб-формы состоят из набора текстовых полей, кнопок, списков и других элементов управления, которые активизируются щелчком мыши. Технически формы передают данные от пользователя удаленному серверу.

Для получения и обработки данных форм используются языки веб-программирования, такие как Java, Python, PHP, Perl.



До появления HTML5 веб-формы представляли собой набор нескольких элементов:

```
<input type="text">  
<input type="password">  
<input type="submit">
```

Для стилизации форм в разных браузерах приходилось прилагать немало усилий. Кроме того, формы требовали применения JavaScript для проверки введенных данных, а также были лишены специфических типов полей ввода для указания повседневной информации типа дат, адресов электронной почты и URL-адресов.

HTML5-формы решили большинство этих распространенных проблем благодаря наличию новых атрибутов, предоставив возможность изменять внешний вид элементов форм за счет CSS3.

```
1 <input type="button" value="Кнопка" />
2 <br /><br />
3 <label><input type="checkbox" /> Выбор №1</label>
4 <label><input type="checkbox" /> Выбор №2</label>
5 <label><input type="checkbox" /> Выбор №3</label>
6 <br /><br />
7
8 <input type="file" />
9 <br />
10 <input type="hidden" />
11 <br />
12
13 <input type="password" />
14 <br /><br />
15
16
17 <label><input type="radio" /> Выбор №1</label>
18 <label><input type="radio" /> Выбор №2</label>
19 <label><input type="radio" /> Выбор №3</label>
20 <br /><br />
21
22 <input type="reset" />
23 <br /><br />
24
25 <input type="submit" />
26 <br /><br />
27
28 <input type="text" />
29 <br />
```

The diagram illustrates the visual representation of various HTML input types. Blue arrows map the code snippets for checkboxes, a file input, a password input, and a submit button to their respective UI components. Red arrows map the code snippets for a radio input, a reset button, and a text input to their UI equivalents.

Тег **<label>** устанавливает связь между определенной меткой, в качестве которой обычно выступает текст и элементом формы (**<input>**, **<select>**, **<textarea>**). Такая связь необходима, чтобы изменять значения элементов формы при нажатии курсором мыши на текст. Кроме того, с помощью **<label>** можно устанавливать горячие клавиши на клавиатуре и переходить на активный элемент подобно ссылкам.

## <form>

Основу любой формы составляет элемент `<form>...</form>`.

Он не предусматривает ввод данных, так как является контейнером, удерживая вместе все элементы управления формы - поля.

Атрибуты этого элемента содержат информацию, общую для всех полей формы, поэтому в одну форму нужно включать поля, объединенные логически.

Атрибуты:

**accept-charset** – список кодировок символов, которые будут использоваться для отправки формы, например, `<form accept-charset="ISO-8859-1">`

**action** – Обязательный атрибут, который указывает url обработчика формы на сервере, которому передаются данные. Представляет из себя файл (например, `action.php`), в котором описано, что нужно делать с данными формы.

В случае, если вся работа будет выполняться на стороне клиента сценариями JavaScript, то для атрибута `action` можно указать значение `#`.

Также можно сделать так, чтобы заполненная посетителем форма приходила вам на почту. Для этого нужно внести следующую запись:

```
<form action="mailto:адрес вашей электронной почты"  
enctype="text/plain"></form>
```

**autocomplete** Отвечает за запоминание введенных в текстовое поле значений и автоподстановку их при последующем вводе:

`on` – означает, что поле не защищено, и его значение можно сохранять и извлекать,  
`off` – отключает автозаполнение для полей форм.

`enctype` – Используется для указания **MIME**-типа данных (Multipurpose Internet Mail Extensions, многоцелевые расширения интернет-почты), отправляемых вместе с формой, например, `enctype="multipart/form-data"`.

Указывается только в случае `method="post"`.

`application/x-www-form-urlencoded` – тип содержимого по умолчанию, указывает на то, что передаваемые данные представляют список URL-кодированных переменных формы.

Символы пробела (ASCII 32) будут закодированы как `+`, а специальный символ, например, такой как `!` будет закодирован шестнадцатиричной форме как `%21`.

`multipart/form-data` – используется для отправки форм, содержащих файлы, не-ASCII данные и бинарные данные, состоит из нескольких частей, каждая из которых представляет содержимое отдельного элемента формы.

`text/plain` – указывает на то, что передается обычный (не `html`) текст.

`name` – Задает имя формы, которое будет использоваться для доступа к элементам формы через сценарии, например, `name="opros"`.

method

Задает способ передачи данных формы.

Метод `get` передает данные на сервер через адресную строку браузера. При формировании запроса к серверу все переменные и их значения формируют последовательность вида `www.anysite.ru/form.php?var1=1&var2=2`.

Имена и значения переменных присоединяются к адресу сервера после знака `?` и разделяются между собой знаком `&`. Все специальные символы и буквы, отличные от латинских, кодируются в формате `%nn`, пробел заменяется на `+`. Этот метод нужно использовать, если вы не передаете больших объемов информации. Если вместе с формой предполагается отправка какого-либо файла, этот метод не подойдет.

Метод `post` применяется для пересылки данных больших объемов, а также конфиденциальной информации и паролей. Данные, отправляемые с помощью этого метода, не видны в заголовке URL, так как они содержатся в теле сообщения.

```
<form action="action.php" enctype="multipart/form-data"  
method="post"></form>
```

`novalidate` Отключает проверку в кнопке для отправки формы. Атрибут используется без указания значения

`target` Указывает окно, в которое будет направлена информация:

`blank` – новое окно

`self` – тот же фрейм

`parent` – родительский фрейм (если он существует, если нет – то в текущий)

`top` – окно верхнего уровня по отношению к данному фрейму. Если вызов происходит не из дочернего фрейма, то в тот же фрейм.

```
<form action="/action_page.php" method="get" target="_blank">
    First name: <input type="text" name="fname"><br>
    Last name: <input type="text" name="lname"><br>
    <input type="submit" value="Submit">
</form>
```

## Группировка элементов формы

Элемент `<fieldset>...</fieldset>` предназначен для группировки элементов, связанных друг с другом, разделяя таким образом форму на логические фрагменты.

Каждой группе элементов можно присвоить название с помощью элемента `<legend>`, который идет сразу за тегом `<fieldset>`. Название группы проявляется слева в верхней границе `<fieldset>`. Например, если в элементе `<fieldset>` хранится контактная информация:

```
<form>
  <fieldset>
    <legend>Контактная информация</legend>
    <p><label for="name">Имя <em>*</em></label><input type="text" id="name"></p>
    <p><label for="email">E-mail</label><input type="email" id="email"></p>
  </fieldset>
  <p><input type="submit" value="Отправить"></p>
</form>
```

The diagram shows a form structure. On the left, the HTML code is displayed. On the right, the resulting user interface is shown. The `<legend>` element is highlighted in blue and has a callout pointing to its representation in the UI as "Контактная информация". The `<input type="text" id="name">` and `<input type="email" id="email">` elements are also highlighted in blue and have callouts pointing to their respective text input fields in the UI. The `<label>` elements are also highlighted in blue and have callouts pointing to the labels "Имя \*" and "E-mail" in the UI. The `<input type="submit" value="Отправить">` element is highlighted in blue and has a callout pointing to its representation in the UI as "Отправить". The `<input type="text" id="name">` and `<input type="email" id="email">` elements are positioned side-by-side within a single `<fieldset>` block, which is visually represented by a light gray border around both inputs.

Атрибут	Значение / описание
disabled	Если атрибут присутствует, то группа связанных элементов формы, находящихся внутри контейнера <code>&lt;fieldset&gt;</code> , отключены для заполнения и редактирования. Используется для ограничения доступа к некоторым полям формы, содержащих ранее введенные данные. Атрибут используется без указания значения – <code>&lt;fieldset disabled&gt;</code> .
form	Значение атрибута должно быть равно атрибуту <code>id</code> элемента <code>&lt;form&gt;</code> в этом же документе. Указывает на одну или несколько форм, к которым принадлежит данная группа элементов. На данный момент атрибут не поддерживается ни одним браузером.
name	Определяет имя, которое будет использоваться для ссылки на элементы в JavaScript, или для ссылки на данные формы после заполнения и отправки формы. Является аналогом атрибута <code>id</code> .

## Поля формы

Элемент `<input>` создает большинство полей формы. Атрибуты элемента отличаются в зависимости от типа поля, для создания которого используется этот элемент.

С помощью CSS-стилей можно изменить размер шрифта, тип шрифта, цвет и другие свойства текста, а также добавить границы, цвет фона и фоновое изображение. Ширина поля задается свойством `width`.

The screenshot shows a 'Shipping Address' form with the following fields:

- First Name: A text input field containing the letter 'I'.
- Last Name: An empty text input field.
- Address: A text input field.
- Apt: An empty text input field.
- City: An empty text input field.
- State: A dropdown menu showing a list of states.
- ZIP Code: An empty text input field.
- Email address for receipt: An empty text input field.
- Primary Phone: An empty text input field.

At the bottom left is a yellow 'CONTINUE' button. The background features a faint watermark of a world map.

**accept-charset** Устанавливает кодировку, в которой сервер может принимать и обрабатывать данные.

**action** Адрес программы или документа, который обрабатывает данные формы.

**autocomplete** Включает автозаполнение полей формы.

**enctype** Способ кодирования данных формы.

**method** Метод протокола HTTP.

**name** Имя формы.

**novalidate** Отменяет встроенную проверку данных формы на корректность ввода.

**target** Имя окна или фрейма, куда обработчик будет загружать возвращаемый результат.

<https://webref.ru/course/html-content/forms>

# type

`button` — создает кнопку.

Кнопка

`checkbox` — превращает поле ввода во флажок, который можно установить или очистить, например,

У меня есть автомобиль

`color` — генерирует палитры цветов в поддерживающих браузерах, давая пользователям возможность выбирать значения цветов в шестнадцатеричном формате.



`date` — позволяет вводить дату в формате дд.мм.гггг.

День рождения:

`datetime-local` — позволяет вводить дату и время, разделенные прописной английской буквой `T` по шаблону дд.мм.гггг чч:мм.

День рождения — день и время:

`email` — браузеры, поддерживающие данный атрибут, будут ожидать, что пользователь введет данные, соответствующие синтаксису адресов электронной почты.

E-mail:

`file` — позволяет загружать файлы с компьютера пользователя.

Выберите файл:  Файл не выбран

`hidden` — скрывает элемент управления, который не отображается браузером и не дает пользователю изменять значения по умолчанию.

`password` — создает текстовые поля в форме, при этом вводимые пользователем символы заменяются на звездочки, маркеры, либо другие, установленные браузером значки.

Введите пароль:

`radio` — создает переключатель — элемент управления в виде небольшого кружка, который можно включить или выключить.

Вегетарианец:

`range` — позволяет создать такой элемент интерфейса, как ползунок, `min / max` — позволят установить диапазон выбора

`reset` — создает кнопку, которая очищает поля формы от введенных пользователем данных.

`search` — обозначает поле поиска, по умолчанию поле ввода имеет прямоугольную форму.

Поиск:

`submit` — создает стандартную кнопку, активизируемую щелчком мыши. Кнопка собирает информацию с формы и отправляет ее для обработки.

`text` — создает текстовые поля в форме, выводя однострочное текстовое поле для ввода текста.

`time` — позволяет вводить время в 24-часовом формате по шаблону чч:мм. В поддерживающих браузерах оно отображается как элемент управления в виде числового поля ввода со значением, изменяемым с помощью мыши, и допускает ввод только значений времени.

Укажите время:

`url` — поле предназначено для указания URL-адресов.

Главная страница:

## Поля ввода (текстовые)

### Элемент

`<textarea>...</textarea>` используется вместо элемента `<input type="text">`, когда нужно создать большие текстовые поля. Текст, отображаемый как исходное значение, помещается внутри тега.

Размеры поля устанавливаются при помощи атрибутов `cols` - размеры по горизонтали, `rows` - размеры по вертикали. Высоту поля можно задать свойством `height`. Все размеры считаются исходя из размера одного символа моноширинного шрифта.

<code>autofocus</code>	Устанавливает фокус на нужном начальном текстовом поле автоматически.
<code>cols</code>	Устанавливает ширину через количество символов. Если пользователь вводит больше текста, появляется полоса прокрутки.
<code>disabled</code>	Отключает возможность редактирования и копирования содержимого поля.
<code>form</code>	Значение атрибута должно быть равно значению атрибута <code>id</code> элемента <code>&lt;form&gt;</code> в этом же документе. Определяет одну или несколько форм, которым принадлежит данное текстовое поле.
<code>maxlength</code>	Значение атрибута задает максимальное число символов для ввода в поле.
<code>name</code>	Задает имя текстового поля.
<code>placeholder</code>	Определяет короткую текстовую подсказку, которая описывает ожидаемое вводимое значение.
<code>readonly</code>	Отключает возможность редактирования содержимого поля.
<code>required</code>	Выводит сообщение о том, что данное поле является обязательным для заполнения.
<code>rows</code>	Указывает число, которое означает, сколько строк должно отображаться в текстовой области.
<code>wrap</code>	Определяет, должен ли текст сохранять переносы строк при отправке формы. Значение <code>hard</code> сохраняет перенос, а значение <code>soft</code> не сохраняет. Если используется значение <code>hard</code> , то должно указываться значение атрибута <code>cols</code> .

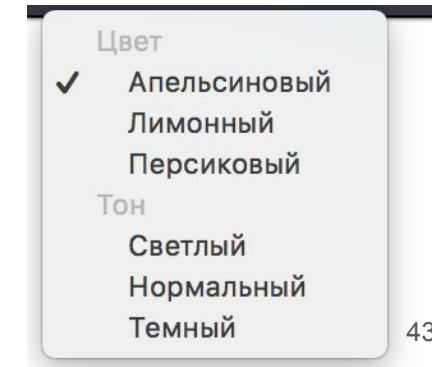
## Раскрывающийся список

Списки дают возможность расположить большое количество пунктов компактно. Раскрывающиеся списки создаются при помощи элемента `<select>...</select>`. Они позволяют выбрать одно или несколько значений из предложенного множества. По умолчанию в поле списка отображается его первый элемент.

Для добавления в список пунктов используются элементы `<option>...</option>`, которые располагаются внутри `<select>`.

Для систематизации списков применяется элемент `<optgroup>...</optgroup>`, который создает заголовки в списках.

Для списков возможно изменить размер шрифта, тип шрифта, цвет и другие свойства текста, а также добавить границы, цвет фона и фоновое изображение.



## select атрибуты

Атрибут	Значение / описание
autofocus	Устанавливает автоматический фокус на элементе при загрузке страницы.
disabled	Отключает раскрывающийся список.
form	Определяет форму, которой принадлежит данный список. В качестве значения атрибута указывается идентификатор формы.
multiple	Дает возможность выбора одного или нескольких пунктов, для этого при выборе нужно нажать и удерживать нажатой клавишу <code>Ctrl</code> .
name	Определяет имя для выпадающего списка. Значение атрибута содержит название, отражающее тематику списка.
required	Выводит сообщение о том, что пользователь должен выбрать значение из раскрывающегося списка перед отправкой формы.
size	Задает количество одновременно видимых на экране элементов списка. Если количество элементов списка превышает установленное количество, появляется полоса прокрутки. Значение атрибута задается целым положительным числом.

## option атрибуты

Атрибут	Значение / описание
disabled	Делает недоступным для выбора элемент списка.
label	Задает укороченную версию для элемента, которая будет отражаться в выпадающем списке. Значение атрибута содержит текст, описывающий соответствующий пункт выпадающего списка.
selected	Отображает выбранный элемент списка по умолчанию при загрузке страницы браузером.
value	Указывает значение, которое будет отправлено на сервер при отправке формы.

## optgroup атрибуты

Атрибут	Значение / описание
disabled	Отключает данную группу элементов списка для выбора.
label	Задает заголовок для группы элементов выпадающего списка. Значение атрибута содержит текст, недоступный для выбора, который будет располагаться над соответствующими пунктами списка. Текст выделяется в браузере жирным начертанием.

# Надписи полей формы

Надписи к элементам формы создаются с помощью элемента `<label>...</label>`.

Существует два способа группировки надписи и поля. Если поле находится внутри элемента `<label>`, то атрибут `for` указывать не нужно.

```
<!-- с указанием атрибута for -->
<label for="comments">Когда вы последний раз летали на самолете?</label>
<textarea id="comments"></textarea>

<!-- без атрибута for -->
<p><label>Кошка<input id="cat" type="checkbox"></label></p>
```

Атрибут	Значение / описание
<code>for</code>	Определяет, к какому полю формы привязан данный элемент. Можно создавать поясняющие надписи к следующим элементам формы: <code>&lt;input&gt;</code> , <code>&lt;textarea&gt;</code> , <code>&lt;select&gt;</code> . Значение атрибута содержит идентификатор поля формы.

## Кнопки

Элемент `<button>...</button>` создает кликабельные кнопки. В отличие от кнопок, созданных `<input>` (`<input type="submit"></input>`, `<input type="image">`, `<input type="reset">`, `<input type="button">`), внутри элемента `<button>` можно поместить контент – текст или изображение.

Для корректного отображения элемента `<button>` разными браузерами нужно указывать атрибут `type`, например, `<button type="submit"></button>`.

Кнопки позволяют пользователям передавать данные в форму, очищать содержимое формы или предпринимать какие-либо другие действия. Можно создавать границы, изменять фон и выравнивать текст на кнопке.

## Флажки и переключатели в формах

Флажки в формах задаются с помощью конструкции `<input type="checkbox">`, а переключатель – при помощи `<input type="radio">`.

Флажков, в отличие от переключателей, в одной форме может быть установлено несколько. Если для флажков указан атрибут `checked`, то при загрузке страницы на соответствующих полях формы флажки уже будут установлены.

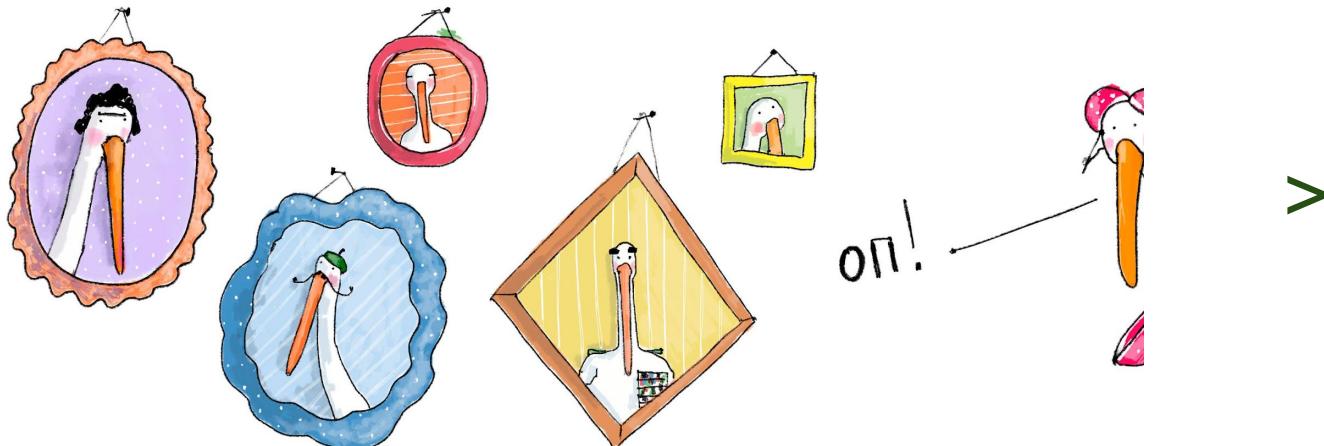
Элемент `<label>` применяется при реализации выбора с помощью переключателей и флажков. Можно выбрать нужный пункт, просто щелкнув мышью на тексте, связанном с ним. Для этого нужно поместить `<input>` внутрь элемента `<label>`.

# HTML-изображения

HTML-изображения добавляются на веб-страницы с помощью тега `<img>`. Использование графики делает веб-страницы визуально привлекательнее. Изображения помогают лучше передать суть и содержание веб-документа.

С помощью HTML-тегов `<map>` и `<area>` можно создавать **карты-изображения** с активными областями.

`<img`



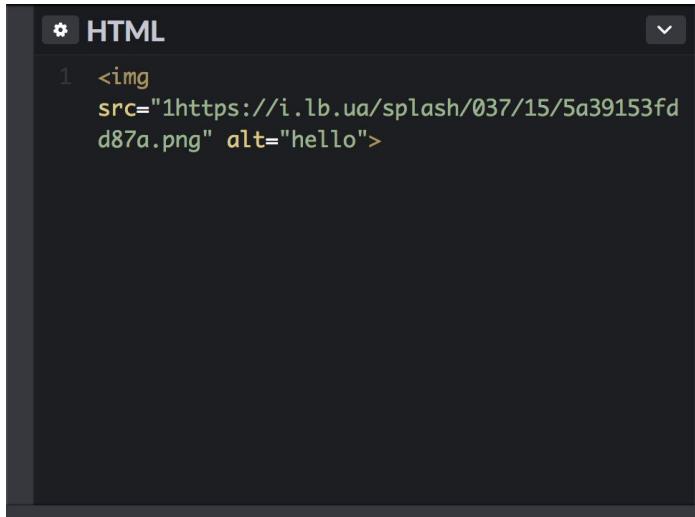
## <img>

Элемент <img> является строчным, то рекомендуется располагать его внутри блочного элемента, например, <p> или <div>.

Тег <img> имеет обязательный атрибут `src`, значением которого является абсолютный или относительный путь к изображению:

```

```



The screenshot shows a browser's developer tools interface with the 'HTML' tab selected. It displays the following code:

```
1 
```

Below the code, there is a small thumbnail image of a green and white graphic, labeled 'hello'.

## Атрибуты

- align** Определяет как рисунок будет выравниваться по краю и способ обтекания текстом.
- alt** Альтернативный текст для изображения.
- border** Толщина рамки вокруг изображения.
- height** Высота изображения.
- hspace** Горизонтальный отступ от изображения до окружающего контента.
- ismap** Говорит браузеру, что картинка является серверной картой-изображением.
- longdesc** Указывает адрес документа, где содержится аннотация к картинке.
- lowsrc** Адрес изображения низкого качества.
- src** Путь к графическому файлу.
- vspace** Вертикальный отступ от изображения до окружающего контента.
- width** Ширина изображения.
- usemap** Ссылка на тег `<map>`, содержащий координаты для клиентской карты-изображения.

```
<img align="bottom | left | middle | right | top">
```

Значение	Описание	Пример
bottom	Выравнивание нижней границы изображения по окружающему тексту..	Lorem ipsum dolor sit amet, consectetur  adipiscing elit...
left	Выравнивает изображение по левому краю окна.	 Lorem ipsum dolor sit amet, consectetur adipiscing elit...
middle	Выравнивание середины изображения по базовой линии текущей строки.	Lorem ipsum dolor sit amet,  consectetur adipiscing elit...
right	Выравнивает изображение по правому краю окна.	Lorem ipsum dolor sit amet, consectetur  adipiscing elit...
top	Верхняя граница изображения выравнивается по самому высокому элементу текущей строки.	Lorem ipsum dolor sit amet,  consectetur adipiscing elit...

## Адрес изображения

Адрес изображения может быть абсолютным URL: `url(http://anysite.ru/images/anyphoto.png)`

Или же через относительный путь от документа или корневого каталога сайта:

`url(..../images/anyphoto.png), url(/images/anyphoto.png).`

Это интерпретируется следующим образом:

`..` — означает подняться вверх на один уровень, к корневому каталогу,

`images/` — перейти к папке с изображениями,

`anyphoto.png` — указывает на файл изображения.

## Размеры изображения

Без задания размеров изображения рисунок отображается на странице в реальном размере. Отредактировать размеры изображения можно с помощью атрибутов `width` и `height`.

!Если будет задан только один из атрибутов, то второй будет вычисляться автоматически для сохранения пропорций рисунка.

## Форматы графических файлов

JPEG (*Joint Photographic Experts Group*)

GIF (*Graphics Interchange Format*)

PNG (*Portable Network Graphics*)

APNG (*Animated Portable Network Graphics*)

SVG (*Scalable Vector Graphics*)

BMP (*Bitmap Picture*)

ICO (*Windows icon*)



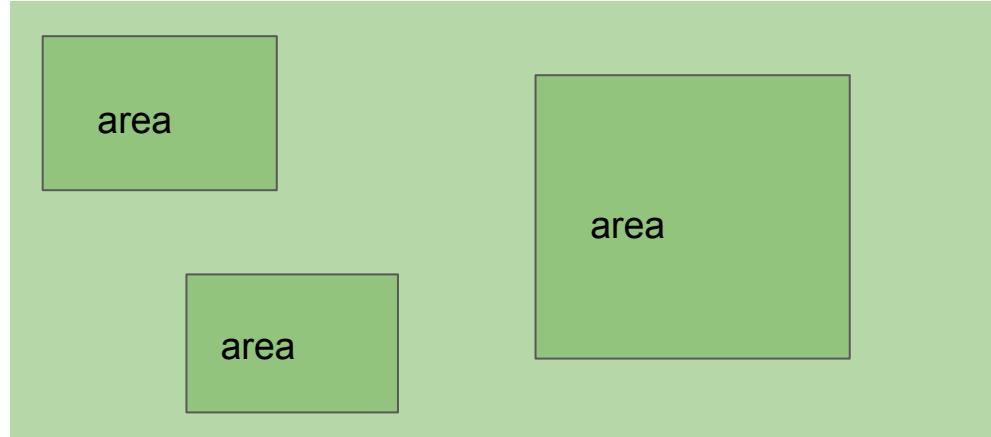
## <map>

Тег `<map>` служит для представления графического изображения в виде карты с активными областями.

Для тега доступен атрибут `name`, который задает имя карты. Значение атрибут `name` для тега `<map>` должно соответствовать имени в атрибуте `usemap` элемента `<img>`

```
  
<map name="имя_карты">  
...  
</map>
```

map



## <area>

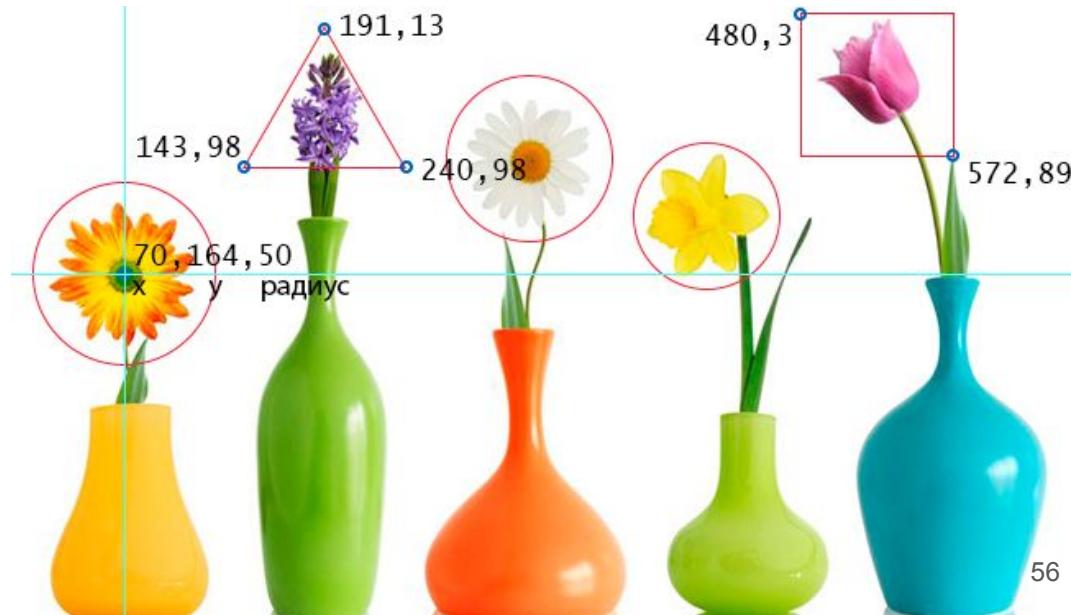
Тег `<area>` описывает только одну активную область в составе карты изображений на стороне клиента. Элемент не имеет закрывающего тела. Если одна активная область перекрывает другую, то будет реализована первая ссылка из списка областей.

```
<map name="имя_карты">  
  <area атрибуты>  
</map>
```

Пример

<https://jsfiddle.net/adamschwartz/Rvz6e/>

<http://jsfiddle.net/jamietre/jQG48/>



## <audio>

Работа с аудио контентом.

До недавнего времени единственным способом добавления звуковых файлов на веб-страницы было интегрирование фонового звука с помощью тега `<bgsound>`, который проигрывался во время просмотра пользователем страницы без возможности выключения.



```
<audio src="name.ogg" controls></audio>
```



```
<audio controls>
  <source src="name.ogg" type="audio/ogg">
  <source src="name.mp3" type="audio/mpeg">
  <a href="sounds/name.mp3">Скачать name.mp3</a>
</audio>
```

Рекомендуется включать несколько источников звука, представленных с использованием атрибута `src` элемента `<source>`. Одновременно можно добавить резервный контент для браузеров, которые не поддерживают элемент `<audio>`.

<code>autoplay</code>	Автоматическое воспроизведение аудио файла сразу же после загрузки страницы.
<code>controls</code>	Указывает браузеру, что нужно отобразить базовые элементы управления воспроизведением (начинать и останавливать воспроизведение, переходить в другое место записи, регулировать громкость).
<code>loop</code>	Циклическое воспроизведение аудио файла.
<code>muted</code>	Выключает звук при воспроизведении аудио файла.
<code>preload</code>	Атрибут, отвечающий за предварительную загрузку аудио контента. Не является обязательным, некоторые браузеры игнорируют его. Возможные значения: <code>auto</code> – браузер загружает аудио файл полностью, чтобы он был доступен, когда пользователь начнет его воспроизведение. <code>metadata</code> – браузер загружает первую небольшую часть аудио файла, чтобы определить его основные характеристики. <code>none</code> – отсутствие автоматической загрузки аудио файла.
<code>src</code>	Содержит абсолютный или относительный URL-адрес аудио файла.

# HTML video

<object> -> <video></video>

Firefox



Google, Opera



Internet Explorer



```
<video src="video.ogv" controls></video>
```

Атрибут `controls` отвечает за появление элементов управления видеоплеером.

Вы можете добавить изображение с помощью атрибута `poster`, которое браузер будет использовать, пока загружается видео или пока пользователь не нажмет на кнопку воспроизведения, а также задать высоту и ширину видео.

Как и в случае с аудиофайлами, рекомендуется перечислять в `<source>` все форматы, начиная с более предпочтительного.

```
<video controls width="400" height="300">
  <source src="video.mp4" type="video/mp4">
  <source src="video.webm" type="video/webm">
  <source src="video.ogv" type="video/ogg">
  <object data="video.swf" type="application/x-shockwave-flash">
    <param name="movie" value="video.swf">
  </object>
</video>
```

## Встраиваемый интерактивный контент

Элемент `<embed>` определяет контейнер для внешнего приложения или интерактивного содержимого (другими словами, плагина). Большинство браузеров поддерживало данный элемент на протяжении долгого времени. Тем не менее, данный тег не был включен в спецификацию HTML4, его добавили в спецификацию HTML5. С помощью `<embed>` на веб-страницы можно добавлять не только видеофайлы, но и векторные изображения в формате `swf`

```
<embed src="movie.mov" type="video/quicktime" width="640" height="480">
<embed src="helloworld.swf">
```

Атрибут	Описание, принимаемое значение
<code>height</code>	Определяет высоту встраиваемого контента в <code>px</code> или <code>%</code> .
<code>src</code>	Содержит абсолютный или относительный URL-адрес медиафайла.
<code>type</code>	Определяет MIME-тип встраиваемого файла.
<code>width</code>	Определяет ширину встраиваемого контента в <code>px</code> или <code>%</code> .

## Font icons



# Font Awesome CDN

Набор шрифтов и значков, основанный на CSS и LESS. Это было сделано Дэйвом Ганди для использования с Twitter Bootstrap, а позже было включено в BootstrapCDN.

## Использование CSS

```
<link rel="stylesheet" href="path/to/font-awesome/css/font-awesome.min.css">
```



## Использование Sass или Less

```
@fa-font-path: "../font";
```

```
<i class="fa fa-camera-retro fa-lg"></i> fa-lg  
<i class="fa fa-camera-retro fa-2x"></i> fa-2x  
<i class="fa fa-camera-retro fa-3x"></i> fa-3x  
<i class="fa fa-camera-retro fa-4x"></i> fa-4x  
<i class="fa fa-camera-retro fa-5x"></i> fa-5x
```

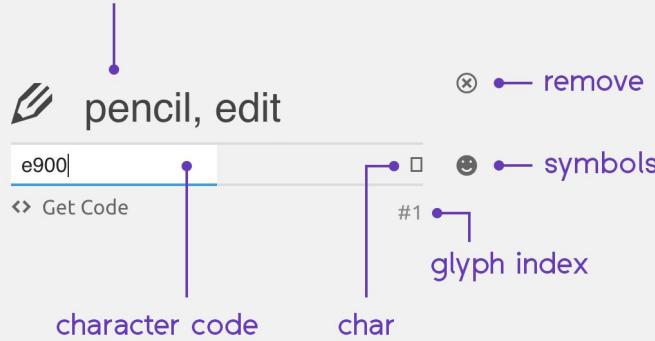
# IcoMoon

IcoMoon - это ещё один популярный иконочный шрифт.

IcoMoon предоставляет онлайн приложение, где вы можете найти и скачать более 3500 иконок. В данном приложении можно осуществлять основных операции: поворот, смену цвета и другое.



comma separated class names  
(without class prefix)



```
@font-face {  
    font-family: 'icomoon';  
    src: url('../fonts/icomoon.eot?58okl');  
    src: url('../fonts/icomoon.eot?58okl#iefix')  
        format('embedded-opentype'),  
    url('../fonts/icomoon.ttf?58okl') format('truetype'),  
    url('../fonts/icomoon.woff?58okl') format('woff'),  
    url('../fonts/icomoon.svg?58okl#icomoon') format('svg');  
    font-weight: normal;  
    font-style: normal;  
}
```

```
.icon-add:before {  
    content: "\e900";  
}  
.icon-arrowdown:before {  
    content: "\e901";  
}  
.icon-arrowpoint:before {  
    content: "\e902";  
}
```

```
[class^="icon-"], [class*=" icon-"] {  
    font-family: 'icomoon' !important;  
    speak: none;  
    font-style: normal;  
    font-weight: normal;  
    font-variant: normal;  
    text-transform: none;  
    line-height: 1;  
}
```

# HTML5

## <canvas>

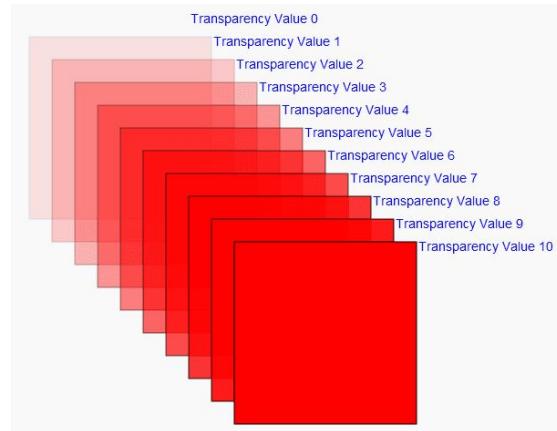
Создает область, в которой при помощи JavaScript можно рисовать разные объекты, выводить изображения, трансформировать их и менять свойства. При помощи тега `<canvas>` можно создавать рисунки, анимацию, игры и др.

Используется, как правило, для отрисовки графиков для статей и игрового поля в некоторых браузерных играх. Но также может использоваться для встраивания видео в страницу и создания полноценного плеера.

```
<canvas id="идентификатор"></canvas>
```

**height** Задает высоту холста. По умолчанию 300 пикселов.

**width** Задает ширину холста. По умолчанию 150 пикселов.



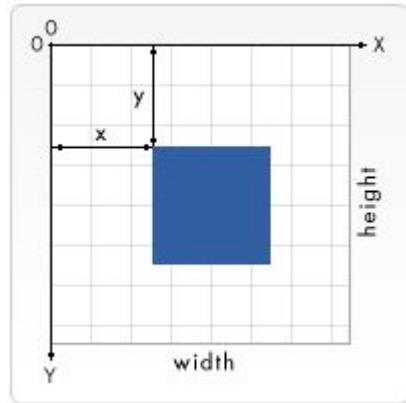
## События

<b>onblur</b>	Потеря фокуса.	<b>onmousemove</b>	Перемещение курсора мыши.
<b>onchange</b>	Изменение значения элемента формы.	<b>onclick</b>	Щелчок левой кнопкой мыши на элементе.
<b>ondblclick</b>	Двойной щелчок левой кнопкой мыши на элементе.	<b>onfocus</b>	Получение фокуса
<b>onkeydown</b>	Клавиша нажата, но не отпущена.	<b>onkeypress</b>	Клавиша нажата и отпущена.
<b>onkeyup</b>	Клавиша отпущена.	<b>onload</b>	Документ загружен.
<b>onmousedown</b>	Нажата левая кнопка мыши.	<b>onmouseout</b>	Курсор покидает элемент.
<b>onmouseover</b>	Курсор наводится на элемент	<b>onmouseup</b>	Левая кнопка мыши отпущена.
<b>onreset</b>	Форма очищена	<b>onselect</b>	Выделен текст в поле формы.
<b>onsubmit</b>	Форма отправлена	<b>onunload</b>	Закрытие окна.

Изменение высоты или ширины холста сотрет всё его содержимое и все настройки, проще говоря он создаётся заново;

Начало отсчёта (точка 0,0) находится в левом верхнем углу. Но её можно сдвигать;

Цвет текста можно указывать аналогично CSS, как и размер шрифта.





1. В отличие от SVG гораздо удобнее иметь дело с большим числом элементов;
  2. Имеет аппаратное ускорение;
  3. Можно манипулировать каждым пикселям;
  4. Можно применять фильтры обработки изображений; Есть много библиотек.
- 
1. Чрезмерно нагружает процессор и оперативную память;
  2. Из-за ограничения сборщика мусора нет возможности очистить память;
  3. Необходимо самому обрабатывать события с объектами;
  4. Плохая производительность при высоком разрешении;
  5. Приходится отрисовывать отдельно каждый элемент.
  6. Возможность создания на страницах специальных «маячков», т.н. Canvas Fingerprinting, для отслеживания пользователей в сети.

```
<!doctype html>
<html lang="ru">
  <head>
    <title>canvas</title>
    <script src="example.js"></script>
  </head>
  <body>
    <canvas id="canvas">Этот элемент не поддерживается</canvas>
  </body>
</html>
```

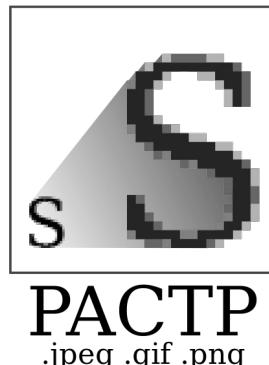
```
function onLoadHandler() {
  var canvas = document.getElementById('canvas'),
      context = canvas.getContext('2d');
  /*
   Далее какие-либо действия над холстом
  */
}
window.onload = onLoadHandler;
```

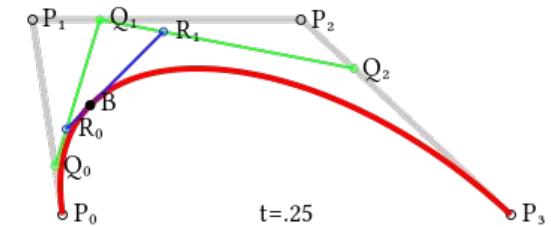


SVG (Scalable Vector Graphics — масштабируемая векторная графика) — язык разметки масштабируемой векторной графики, созданный (W3C) и входящий в подмножество расширяемого языка разметки XML.

Поддерживает как неподвижную, так и анимированную интерактивную графику — или, в иных терминах, декларативную и скриптовую.

В основу SVG легли языки разметки VML и PGML. Разрабатывается с 1999 года. В 2001 году вышла версия 1.0, в 2011 — версия 1.1, которая остаётся актуальной до сегодняшнего дня. В настоящее время в активной разработке находится версия 2.





## Возможности языка

Описание путей. Позволяет задать любую фигуру компактной строкой, описывающей путь от начальной точки до конечной через любые промежуточные координаты.

```
<path fill="none"  
stroke="black" d="M 227 239 L 328 90 L 346 250 L 201 124 L 410 150 L 228 238" />
```

Описание основных геометрических фигур (многоугольники, прямоугольники, окружности и т. п.).

Широкий спектр визуальных свойств, которые можно применить к фигурам и путям: окраска, прозрачность, скругление углов и т. д.

Интерактивность. На каждый отдельный элемент и на целое изображение можно повесить обработчик событий (клик, перемещение, нажатие клавиши и т.д), таким образом, пользователь может управлять рисунком (например — перемещать мышкой некоторые элементы).

Анимация и сценарии. С помощью ECMAScript или JavaScript можно описывать даже самые сложные сценарии, связанные с математическими вычислениями координат и пропорций фигур.

# Структура документа SVG

Первая строка — стандартный XML-заголовок

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
```

Во второй и третьей строках должен располагаться заголовок **DOCTYPE**

```
<!DOCTYPE svg PUBLIC "-//W3C//DTD SVG 1.1//EN"  
      "http://www.w3.org/Graphics/SVG/1.1/DTD/svg11.dtd">
```

В четвёртой строке размещается корневой элемент документа с указанием пространства имён SVG.

```
<svg version="1.1"  
      baseProfile="full"  
      xmlns="http://www.w3.org/2000/svg"  
      xmlns:xlink="http://www.w3.org/1999/xlink"  
      xmlns:ev="http://www.w3.org/2001/xml-events"  
      width="100%" height="100%">
```

```

<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<svg version = "1.1"
      baseProfile="full"
      xmlns = "http://www.w3.org/2000/svg"
      xmlns:xlink = "http://www.w3.org/1999/xlink"
      xmlns:ev = "http://www.w3.org/2001/xml-events"
      height = "400px" width = "400px">
    <rect x="0" y="0" width="400" height="400"
          fill="none" stroke="black" stroke-width="5px" stroke-opacity="0.5"/>
    <g fill-opacity="0.6" stroke="black" stroke-width="0.5px">
      <circle cx="200px" cy="200px" r="104px" fill="red" transform="translate( 0,-52)" />
      <circle cx="200px" cy="200px" r="104px" fill="blue" transform="translate( 60, 52)" />
      <circle cx="200px" cy="200px" r="104px" fill="green" transform="translate(-60, 52)" />
    </g>
</svg>

```

```

<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<svg width="198" height="188"
      viewBox="0 0 198 188"
      version="1.1"
      baseProfile="full"
      xmlns="http://www.w3.org/2000/svg"
      xmlns:xlink="http://www.w3.org/1999/xlink"
      xmlns:ev="http://www.w3.org/2001/xml-events">
    <g id="Page-1" stroke="none" stroke-width="1" fill="none" fill-rule="even-odd">
      <polygon id="Star-1" stroke="#979797" stroke-width="3" fill="#F8E81C"
              points="99 154 40 185 51 119 4 73 69 64 99 3 128 64 194 73 147 119 158 185 ">
    </polygon>
  </g>
</svg>

```

браузер	версия
Internet Explorer	9+[3]
Mozilla Firefox	1.5+[4]
Netscape Navigator	9.0+
Google Chrome	3.0+
Safari	4.0+
Opera	8.0+

**Спасибо за внимание!**