

HTML5
JAVA SCRIPT
JQUERY
CSS3

HTML5

Инструктор: Максим

1. Ссылки
2. Изображения
3. Списки
4. Таблицы
5. Разметка Web-страницы



1. Ссылки

Web-страницы отличаются от своих бумажных аналогов в первую очередь наличием гиперссылок, связывающих документы друг с другом. Для обозначения таких ссылок применяется элемент "a". Адрес страницы (URI), которую должен загрузить браузер при нажатии на ссылку, указывается в атрибуте **href**:

```
<a href="http://www.example.com/">текст ссылки</a>
```

Если язык страницы, на которую ведет ссылка, отличен от языка ссылающейся страницы, то можно добавить атрибут **hreflang**, значением которого должен быть код языка. Также через дефис можно добавить код страны:

```
<a href="http://www.example.com/" hreflang="en-US">  
текст ссылки</a>
```

1. Ссылки. Навигация по Web-странице

Элемент "a" можно применять для ссылок на конкретную часть самого же ссылающегося документа. Для этого перед этой частью необходимо вставить элемент, присвоив ему идентификатор id с уникальным значением. Ссылка добавляется там, где она необходима, указав в атрибуте href значение id необходимой метки и добавив перед ним символ #. При нажатии на такую ссылку страница не будет перезагружена, а лишь изменится позиция скроллера

Метка:

```
<h1 id="top">Заголовок</h1>
```

Ссылка на метку:

```
<p><a href="#top">Вернуться вверх</a></p>
```

1. Ссылки. type

Элемент "a" может указывать не только на HTML-страницы, но и на файлы самых разных типов. При необходимости **MIME-тип** ресурса определяется в атрибуте **type**. Атрибут type должен добавляться только при наличии атрибута href:

```
<a href="Pic.jpg" type="image/jpeg">Ссылка на изображение</a>
```



После нажатия на ссылку, браузер отобразит изображение:



Список MIME-типов: <https://webref.ru/html/value/mime>

MIME (Multipurpose Internet Mail Extensions, многоцелевые расширения интернет-почты) – стандарт, описывающий передачу различных типов данных, а также, спецификация для кодирования информации и форматирования сообщений таким образом, чтобы их можно было пересылать по Интернету

1. Ссылки. Абсолютный и относительный путь

В гиперссылках можно указывать как **абсолютный** (полный), так и **относительный** (сокращенный) путь к ресурсам

Абсолютные URI:

```
http://www.example.com/html/  
https://www.example.com/#info  
mailto:info@example.com
```

Относительные URI:

```
relative/file.txt  
../parent/  
page.html
```

1. Ссылки. Абсолютный и относительный путь

Относительные пути по умолчанию рассчитываются от полного пути к вашему документу. Это правило можно изменить, с помощью элемента **base** с атрибутом href. В нем указывается базовый путь, относительно которого и будут рассчитываться все последующие адреса. Элемент base располагается внутри элемента head. На ссылках и прочих элементах с указанными абсолютными адресами это никак не отразится:

```
<head>
    ...
    <base href="http://www.example.com/" />
</head>
...
<a href="http://www.example.com/contact.html">абсолютный
URI</a>,
<a href="contact.html">относительный URI</a>
```

Обе ссылки указывают на одну и ту же Web-страницу

2. Изображения

Для вставки в HTML-документ изображений предназначен элемент **img**, атрибуты:

- **src** – относительный или абсолютный URI изображения (**обязательный**)
- **alt** – короткое описание (отображается при отключенной графике, **обязательный**)
- **width** и **height** – ширина и высота в пикселях

Если не указаны ширина и высота, то картинка отобразится в оригинальном размере, либо, если графика в браузере отключена, то элемент примет такой размер, чтобы в него поместилась alt-надпись. Указав лишь один из этих атрибутов, получите изображение требуемой ширины или высоты с сохранением оригинальных пропорций. **Обратите внимание**, для достижения точных размеров указывайте оба значения, но помните, что если заданные атрибуты не совпадают с оригиналом изображения, то масштабирование приведет к потере его качества

2. Изображения

Ширину и высоту можно задать только в пикселях, просто указав требуемое числовое значение:

```
  
<br />
```

```
  

```



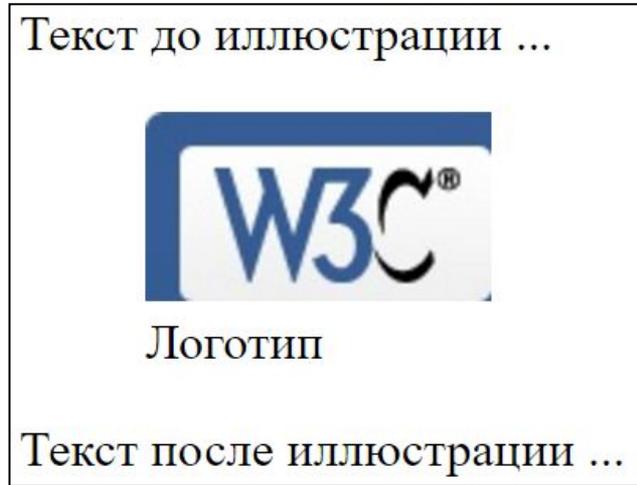
2. Изображения. Подпись. Ссылка

Элемент **figure** применяется для аннотации различных иллюстраций, диаграмм, фотографий и т.д. А элемент **figcaption** просто обортывает заголовок для содержимого внутри элемента figure:

Текст до иллюстрации ...

```
<figure>
  
  <figcaption>Логотип</figcaption>
</figure>
```

Текст после иллюстрации ...



Поместив внутрь элемента "a" элемент img, можно сделать ссылку-изображение:

```
<a href="https://www.w3.org">
  
</a>
```

3. Списки. Ненумерованный

Стандартом HTML предлагается выбор из **трех** типов списков. Все они являются составными конструкциями и формируются с помощью структурных элементов. Первые два типа – это **нумерованный** и **ненумерованный** списки. Обозначаются они элементами **ol** и **ul** соответственно. Пункты списка располагаются внутри этих элементов в элементах **li**, следующих друг за другом:

```
<ul>  
  <li>HTML</li>  
  <li>CSS</li>  
  <li>JS</li>  
</ul>
```

- HTML
- CSS
- JS

3. Списки. Нумерованный

Элементы нумерованного списка, автоматически нумеруются последовательными числами, начиная с единицы. Чтобы изменить стартовое значение, необходимо указать его в атрибуте **start**. В HTML5 появилась возможность изменить направления счета на обратное с помощью атрибута **reversed="reversed"**:

```
<ol start="15" reversed="reversed">  
  <li>HTML</li>  
  <li>CSS</li>  
  <li>JS</li>  
</ol>
```

15. HTML
14. CSS
13. JS

3. Списки. Термины с определениями

Список определений представляет такой список, который содержит термин и определение к этому термину. Для создания списка определений применяется элемент **dl**. Внутри этого элемента помещаются элементы списка. Каждый элемент списка состоит из **термина** и **определения**. Термин помещается в элемент **dt**, а определение – в элемент **dd**:

```
<dl>  
  <dt>HTML</dt>  
  <dd>HyperText Markup Language</dd>  
  <dt>CSS</dt>  
  <dd>Cascading Style Sheets</dd>  
  <dt>JS</dt>  
  <dd>JavaScript</dd>  
</dl>
```

HTML	HyperText Markup Language
CSS	Cascading Style Sheets
JS	JavaScript

4. Таблицы

Для создания таблиц используется элемент **table**. Каждая таблица содержит строки, который представлены элементом **tr**. А каждая строка содержит ячейки в виде элементов **th** или **td**

Элементы **th** (не обязательный) обозначают ячейки с заголовками столбцов или строк. Содержимое их по умолчанию отображается браузерами жирным начертанием и центрируется

Элементы **td** же предназначены для всех остальных ячеек с данными

Элемент **caption** отображает описание таблицы, находится на самом первом месте внутри элемента **table**

4. Таблицы

```

<table border="1">
  <caption>Выписка лицевого счета</caption>
  <tr>
    <th>Месяц</th>
    <th>Баланс</th>
  </tr>
  <tr>
    <td>Сентябрь</td>
    <td>5000 грн.</td>
  </tr>
</table>

```

Выписка лицевого счета

Месяц	Баланс
Сентябрь	5000 грн.

4. Таблицы. Объединение ячеек

Чтобы таблица имела прямоугольный вид, в каждой строке должно быть одинаковое количество ячеек. Если нужно растянуть ячейку на несколько столбцов или строк используются атрибуты **colspan** и **rowspan** элементов **td** и **th**. Присвоив ячейке атрибут `colspan="3"`, укажите браузеру, что этот элемент занимает место трех ячеек в строке – свое собственное и двух следующих, две следующие ячейки указывать не нужно и строка будет содержать на 2 элемента **td** (или **th**) меньше. Аналогично работает атрибут `rowspan="3"`, только применяется к столбцам

Обратите внимание, если применить один из этих атрибутов, но не **удалить "лишние" ячейки**, то браузер сместит их в следующий столбец, передвинув остальные ячейки еще дальше и **исказив** тем самым таблицу

	colspan="3"					
	rowspan="3"		colspan="3"			
			rowspan="2"			

4. Таблицы. Объединение ячеек

```

<table border="1">
  <tr>
    <td colspan="2">1.1 1.2</td>
    <td>1.2</td> <!-- лишняя ячейка -->
    <td>1.3</td>
  </tr>
  <tr>
    <td>2.1</td>
    <td>2.2</td>
    <td rowspan="2">2.3<br />3.3</td>
  </tr>
  <tr>
    <td>3.1</td>
    <td>3.2</td>
    <td>3.3</td> <!-- лишняя ячейка -->
  </tr>
</table>

```

1.1 1.2	1.2	1.3
2.1	2.2	2.3
3.1	3.2	3.3

4. Таблицы. Объединение ячеек

```

<table border="1">
  <tr>
    <td colspan="2">1.1 1.2</td>
    <td>1.3</td>
  </tr>
  <tr>
    <td>2.1</td>
    <td>2.2</td>
    <td rowspan="2">2.3<br />3.3</td>
  </tr>
  <tr>
    <td>3.1</td>
    <td>3.2</td>
  </tr>
</table>

```

1.1	1.2	1.3
2.1	2.2	2.3
3.1	3.2	3.3

4. Таблицы. Объединение ячеек

```

<table border="1">
  <tr>
    <td>1.1</td>
    <td colspan="2" rowspan="2">1.2 1.3<br />2.2
2.3</td>
    <!-- <td>1.3</td> лишняя ячейка -->
  </tr>
  <tr>
    <td>2.1</td>
    <!-- <td>2.2</td> лишняя ячейка -->
    <!-- <td>2.3</td> лишняя ячейка -->
  </tr>
  <tr>
    <td>3.1</td>
    <td>3.2</td>
    <td>3.3</td>
  </tr>
</table>

```

1.1	1.2 1.3	
2.1	2.2 2.3	
3.1	3.2	3.3

4. Таблицы. Объединение ячеек. Наложение

```
<table border="1">
  <tr>
    <td>1.1</td>
    <td rowspan="2">1.2<br />2.2</td>
  </tr>
  <tr>
    <td colspan="2">2.1 2.2</td>
  </tr>
</table>
```

1.1	1.2
2.1	2.2

4. Таблицы. Группирование строк

Для группирования **строк** используются элементы:

- **thead** – строки с ячейками заголовка
- **tfoot** – строки с "итоговыми" ячейками
- **tbody** – строки с данными

Несмотря на то, что строки из `tfoot` будут отображены последними, размещать сам элемент `tfoot` можно как после последнего, так и перед первым элементом `tbody` (после `thead`). В одной таблице может быть только **по одному элементу `thead` и `tfoot`**, и любое количество `tbody`. Визуально эти теги не отличаются, пока не определить для них соответствующие стили CSS

Обратите внимание, если используется `thead` и/или `tfoot`, то обязательно нужно использовать и `tbody`. Причем в каждой из групп обязательно должна быть хотя бы одна строка `tr`, и не должно быть строк вне групп

4. Таблицы. Группирование строк

```
<table>
  <thead style="background: red;">
    <tr>
      <td>Красный</td>
      <td>Красный</td>
    </tr>
  </thead>
  <tfoot style="background: blue;">
    <tr>
      <td>Синий</td>
      <td>Синий</td>
    </tr>
  </tfoot>
  <tbody style="background: lime;">
    <tr>
      <td>Зеленый</td>
      <td>Зеленый</td>
    </tr>
  </tbody>
</table>
```

Красный	Красный
Зеленый	Зеленый
Синий	Синий

5. Разметка Web-страницы. div. span

Форматирование текста, добавление в документ ссылок, изображений, списков и таблиц недостаточно для создания полноценных Web-страниц, которые обычно состоят из нескольких логически независимых частей. Так, на страницах сайта, помимо основной информации, обычно присутствует верхний колонтитул (шапка), нижний колонтитул (подвал), главное меню и прочие навигационные блоки

HTML4 и более ранние версии языка применялись в первую очередь для визуального оформления документа. Для создания каркаса страницы было достаточно всего двух элементов: блочного `div` и строчного `span`, которые сами по себе не влияют на отображение текста (кроме стандартных "блочных" свойств `div`), но позволяют использовать глобальные атрибуты `class` и `id`. И с их помощью задать стили отображения в CSS. Тем более, многие элементы стандарта HTML4 поддерживали целый набор атрибутов для визуального оформления:

`<!-- атрибут type устанавливает вид маркера списка, в HTML5 не поддерживается -->`

```
<ul type="square">
  <li>Один</li>
  <li>Два</li>
</ul>
```



5. Разметка Web-страницы. style

В HTML5 изменилась сама идеология составления документа. Большинство атрибутов визуального оформления теперь **не поддерживаются**, а большинство элементы предназначены для **семантической разметки**, что привело к необходимости разнообразить список элементов и обновить их значение

Элементы `div` и `span` по-прежнему поддерживаются и могут быть применены в целях стилевого оформления когда семантическая разметка не имеет значения. Кроме того, поддерживается элемент **style**, позволяющий определять стили CSS непосредственно в документе. Элемент `style` располагается внутри `head` и содержит в себе прямые инструкции на языке CSS

5. Разметка Web-страницы. div. span. style

```
<head>
  ...
  <style>
    .status {
      color: red;
    }
    #download {
      font-weight: bold;
    }
  </style>
</head>
<body>
  <div id="download">
    Скачивание файла: <span class="status">завершено</span>
  </div>
</body>
```

Скачивание файла: завершено

5. Разметка Web-страницы. article

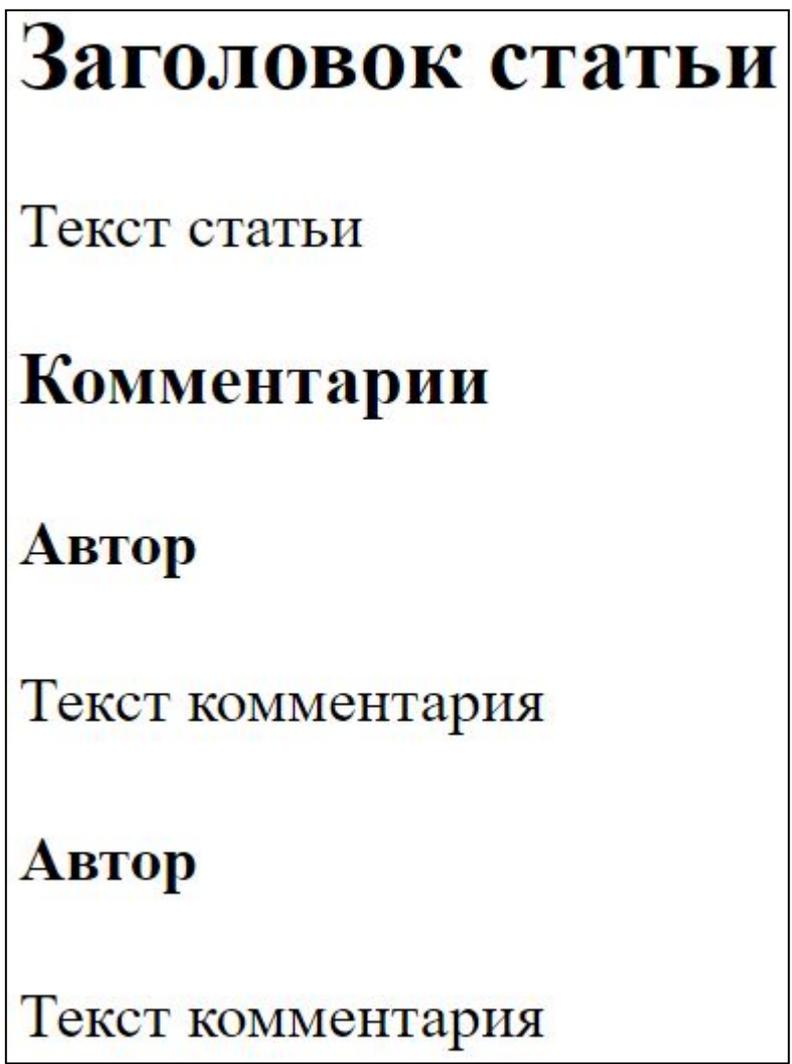
Для разметки Web-страницы HTML5 предлагает несколько элементов с различным семантическим значением

Элемент **article** представляет целостный блок информации на странице, который может рассматриваться отдельно и использоваться независимо от других блоков. Например, это может быть пост на форуме или статья в блоге, онлайн-журнале, комментарий пользователя

Один элемент `article` может включать несколько элементов `article`. Например, можно заключить в элемент `article` всю статью в блоге, и этот элемент будет содержать другие элементы `article`, которые представляют комментарии к этой статье в блоге. То есть статья в блоге может рассматриваться как отдельная семантическая единица, и в то же время комментарии также могут рассматривать отдельно вне зависимости от другого содержимого

Обратите внимание, при использовании `article` надо учитывать, что каждый элемент `article` должен быть идентифицирован с помощью включения в него заголовка `h1-h6`

```
<body>
  <article>
    <h2>Заголовок статьи</h2>
    <div>Текст статьи</div>
    <div>
      <h3>Комментарии</h3>
      <article>
        <h4>Автор</h4>
        Текст комментария
      </article>
      <article>
        <h4>Автор</h4>
        Текст комментария
      </article>
    </div>
  </article>
</body>
```



5. Разметка Web-страницы. section

Элемент `section` объединяет связанные между собой куски информации HTML-документа, выполняя их группировку. Например, `section` может включать набор вкладок на странице, новости, объединенные по категории и т.д.

Допускается вкладывать один элемент `section` внутрь другого

При этом элемент `section` может содержать несколько элементов `article`, выполняя их группировку, так и один элемент `article` может содержать несколько элементов `section`

Обратите внимание, Каждый элемент `section` должен быть идентифицирован с помощью заголовка `h1-h6`

```
<body>  
  <article>  
    <h2>Заголовок статьи</h2>  
    <section>  
      <h3>Содержимое</h3>  
      Текст статьи  
    </section>  
    <section>  
      <h3>Комментарии</h3>  
      <article>  
        <h4>Автор</h4>  
        Текст комментария  
      </article>  
      <article>  
        <h4>Автор</h4>  
        Текст комментария  
      </article>  
    </section>  
  </article></body>
```

Заголовок статьи

Содержимое

Текст статьи

Комментарии

Автор

Текст комментария

Автор

Текст комментария

5. Разметка Web-страницы. nav

Элемент `nav` предназначен для элементов навигации по сайту. Как правило, это нумерованный список с набором ссылок

На одной Web-странице можно использовать несколько элементов `nav`. Например, один элемент навигации для перехода по страницам на сайте, а другой – для перехода внутри HTML-документа

Не все ссылки обязательно помещать в элемент `nav`. Например, некоторые ссылки могут не представлять связанного блока навигации, например, ссылка на главную страницу, на лицензионное соглашение по поводу использования сервиса и подобные ссылки, которые часто помещаются внизу страницы. Как правило, их достаточно определить в элементе `footer`, а элемент `nav` для них использовать необязательно

```
<body>
  <h1>Заголовок сайта</h1>
  <nav>
    <ul>
      <li><a href="html.html">HTML</a></li>
      <li><a href="css.html">CSS</a></li>
      <li><a href="js.html">JavaScript</a></li>
    </ul>
  </nav>
  ...
</body>
```

Заголовок сайта

- [HTML](#)
- [CSS](#)
- [JavaScript](#)

...

5. Разметка Web-страницы. header. footer

Верхний и нижний колонтитулы обозначаются элементами **header** и **footer** соответственно. Верхний может содержать заголовок, вводную информацию о документе, навигацию, форму поиска, логотип и прочее. Нижний колонтитул включает информацию об авторе, дату составления, копирайт и прочее:

```
<body>  
  <header>  
    <h1>Заголовок страницы</h1>  
    <p>Верхний колонтитул</p>  
  </header>  
  <div>Тело страницы</div>  
  <footer>  
    <p>Нижний колонтитул</p>  
  </footer>  
</body>
```

Заголовок страницы

Верхний колонтитул

Тело страницы

Нижний колонтитул

Обратите внимание, элементы header и footer могут быть использованы не только в качестве колонтитулов HTML-документа, но и выполнять аналогичную роль для отдельных его разделов, если это необходимо

```
<body>
  <header>
    <h1>Заголовок страницы</h1>
    <p>Верхний колонтитул сайта</p>
  </header>
  <section>
    <header>
      <h2>Заголовок секции</h2>
      <p>Верхний колонтитул секции</p>
    </header>
    <article>
      <header>
        <h3>Заголовок статьи</h3>
        <p>Верхний колонтитул статьи</p>
      </header>
      ...
    </article>
    ...
  </section></body>
```

Заголовок страницы

Верхний колонтитул сайта

Заголовок секции

Верхний колонтитул секции

Заголовок статьи

Верхний колонтитул статьи

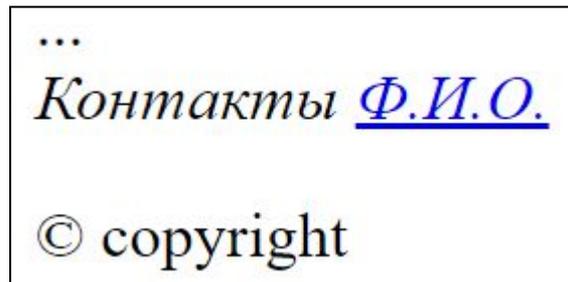
...

...

5. Разметка Web-страницы. address

Элемент **address** предназначен для отображения контактной информации, которая связана с ближайшим элементом `article` или `body`. Нередко данный элемент размещается в футере:

```
<body>
...
<footer>
  <address>
    Контакты <a href="mailto:mail@mail">Ф.И.О.</a>
  </address>
  <p>© copyright</p>
</footer>
</body>
```



5. Разметка Web-страницы. aside

Элемент **aside** представляет содержимое, которое косвенно связано с остальным контентом Web-страницы и которое может рассматриваться независимо от него. Данный элемент можно использовать, например, для сайдбаров, для рекламных блоков, блоков навигационных элементов, различных плагинов и т.д.:

```
<body>  
  <h1>Программирование</h1>  
  <aside>  
    <p>Большинство хороших программистов делают свою работу не  
потому, что ожидают оплаты или признания, а потому что получают  
удовольствие от программирования. (Линус Торвальдс)</p>  
  </aside>  
  <article>  
    <p>В узком смысле (так называемое кодирование) под  
программированием понимается написание инструкций (программ) на  
конкретном языке программирования (часто по уже имеющемуся  
алгоритму – плану, методу решения поставленной задачи).  
Соответственно, люди, которые этим занимаются, называются  
программистами (на профессиональном жаргоне – кодерами), а те,  
кто разрабатывают алгоритмы – алгоритмистами, специалистами  
предметной области, математиками.</p>  
  </article></body>
```

CCS код для предыдущего примера:

```
<style>
p {
    margin: 0;
}
aside {
    background: #f0f0f0;
    text-align: justify;
    padding: 10px;
}
article {
    text-align: justify;
}
</style>
```

Программирование

Большинство хороших программистов делают свою работу не потому, что ожидают оплаты или признания, а потому что получают удовольствие от программирования. (Линус Торвальдс)

В узком смысле (так называемое кодирование) под программированием понимается написание инструкций (программ) на конкретном языке программирования (часто по уже имеющемуся алгоритму – плану, методу решения поставленной задачи). Соответственно, люди, которые этим занимаются, называются программистами (на профессиональном жаргоне – кодерами), а те, кто разрабатывают алгоритмы – алгоритмистами, специалистами предметной области, математиками.

5. Разметка Web-страницы. main

Элемент **main** предназначен для основного содержимого документа. На странице может быть **только один** main и он не должен располагаться внутри элементов article, aside, footer, header или nav:

```
<body>  
  <header>  
    <h1>Заголовок страницы</h1>  
    <p>Верхний колонтитул</p>  
  </header>  
  <main>  
    <p>Основное содержимое</p>  
  </main>  
  <footer>  
    <p>Нижний колонтитул</p>  
  </footer>  
</body>
```

Заголовок страницы

Верхний колонтитул

Основное содержимое

Нижний колонтитул

Не используйте `section` как обёртку для оформления:

```
<!-- Плохой код -->
<body>
  <section class="wrapper">
    <header>
      <h1>Заголовок</h1>
      Содержимое
    </header>
    <section class="main">
      Содержимое страницы
    </section>
    <section class="secondary">
      Вторичное содержимое
    </section>
    <footer>
      Содержимое
    </footer>
  </section>
</body>
```

```
<!-- Хороший код -->
<body>
  <header>
    <h1>Заголовок</h1>
    Содержимое
  </header>
  <div>
    Содержимое страницы
  </div>
  <aside>
    Вторичное содержимое
  </aside>
  <footer>
    Содержимое
  </footer>
</body>
```

Если header содержит только один заголовочный элемент, то он не нужен:

```
<!-- Плохой код -->  
<article>  
  <header>  
    <h1>Заголовок</h1>  
  </header>  
  Содержимое  
</article>
```

```
<!-- Хороший код -->  
<article>  
  <h1>Заголовок</h1>  
  Содержимое  
</article>
```



<http://webref.ru/>

<http://professorweb.ru/>

Спасибо за внимание!