HTML, основы верстки

HTML: Структура документа

1. Декларация версии HTML
2. Мета-информация о странице
3. Тело документа

HTML: DOCTYPE

предназначен для указания типа текущего документа — Document Type Definition (DTD)

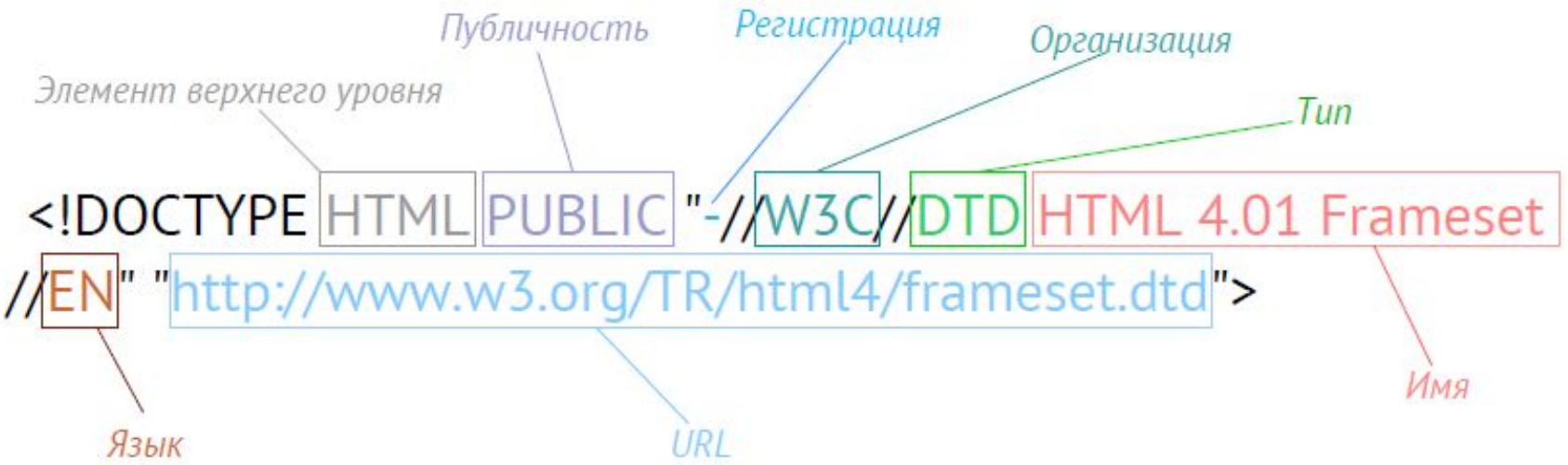
```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
"http://www.w3.org/TR/html4/strict.dtd">

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01
Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Frameset//EN"
"http://www.w3.org/TR/html4/frameset.dtd">

<!DOCTYPE html>
```

HTML: DOCTYPE



HTML: документ

```
<!DOCTYPE html>
<head>
   <title>Шаблон HTML документа</title>
   <meta name="keywords" content="###">
   <meta name="description" content="###">
</head>
<body>
   <p>Параграф.</p>
</body>
</html>
```

Особенности

перемешивание текста документа с инструкциями разметки в потоке данных или файле:

```
<h1> HTML </h1>
<p>
<em>HTML</em> (от англ. HyperText Markup
Language — «язык разметки гипертекста») —
стандартный <i>язык разметки документов</i>
во Всемирной паутине. Большинство веб-страниц
создаются при помощи языка HTML (или XHTML).
</p>
```

Форматирование текста (блочные элементы)

- Занимают всю доступную ширину, высота элемента определяется его содержимым, и он всегда начинается с новой строки.
- Заголовки <h1></h1>, ... ,<h6></h6>
- Параграфы
- Универсальные блочные элементы
- Выделение длинных цитат
- <blockquote></blockquote>

HTML: заголовки

```
<h1>Заголовок первого уровня</h1>
<h2>Заголовок второго уровня</h2>
<h3>Заголовок третьего уровня</h3>
<h4>Заголовок четвертого уровня</h4>
<h5>Заголовок пятого уровня</h5>
<h6>Заголовок шестого уровня</h6>
<h7>Заголовок седьмого уровня</h7>
```

Блочные элементы (пример)

- <body>
-
- Параграф в несколько строк: Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Lorem ipsum dolor sit amet, consectetuer adipiscing elit.
-
-
- Универсальный блочный элемент
-
- <blockquote>Длинная цитата внутри документа: Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diem nonummy nibh euismod tincidunt ut lacreet dolore magna aliguam erat volutpat.
- </blockquote>
- </body>

HTML: текст

```
<p>Абзац</p>
<br> - новая строка
<hr> - горизонтальная линия
<i>курсив</i>
<b>жирный</b>
<u>подчеркивание</u>
<s>зачеркивание</s>
<sub>подстрочный текст</sub>
<sup>надстрочный текст</sup>
```

HTML: списки

```
<UL>
    <LI> первый элемент </LI>
    <LI> второй элемент </LI>
    <LI> третий элемент </LI>
</UL>

<OL>
    <LI> первый элемент </LI>
    <LI> второй элемент </LI>
    <LI> третий элемент </LI>
</OL>
```

HTML: таблицы

```
<table>
  <tr>
    <td>AA</td><td>BB</td>
  </tr>
  <tr>
    <td>CC</td><td>DD</td>
  </tr>
</table>
```

HTML: таблицы

```
<table
border="1"
cellspacing="0"
width="100%" >
```

HTML: таблицы

```
<td
   colspan=3
align=left
width=100%>sss</td>
```

HTML формы

- предназначены для организации интерактивного обмена информацией между пользователем и web-приложением.
- Контейнер формы <form></form>
- Когда форма отправляется на сервер, управление данными передается web-приложению, заданному атрибутом action="URL"

Элементы управления HTML формы

- Служат для взаимодействия пользователя с формой.
- Большинство элементов ввода и управления в форме можно описать при помощи элемента <input>, обязательными атрибутами которого являются:
- name — приписывает данному элементу ввода уникальное имя, использующееся для дальнейшей обработки формы
- type — определяет тип элемента управления или ввода.
- Подписи и метки элементов управления задаются с помощью элемента <label>

Элементы интерфейса формы

- Атрибут type тега <input> позволяет задавать:
  - текстовое поле (text)
  - поле с паролем (password)
  - переключатель (radio)
  - флажок (checkbox)
  - скрытое поле (hidden)
  - кнопку (button)
  - кнопку для отправки формы (submit)
  - кнопку для очистки формы (reset)
  - поле для отправки файла (file)
  - кнопку с изображением (image)

Фрейм

отображает содержимое другой веб - страницы, адрес которой прописывался в параметрах фрейма. Таким образом, веб-страница состояла из набора фреймов, при этом подгружается только требуемое содержимое. Структуру фреймов, правда, при этом поменять нельзя.

Стили верстки веб-страниц

1. Текстовый;
2. Фреймовый;
3. Табличный;
4. Блочный.

Основной минус текстового и табличного стилей верстки это то, что общие элементы для всех страниц сайта не подгружаются, а прописываются для каждой страницы

- **Блочная модель CSS**

Блочная модель.

Знание блоковой модели CSS позволяет верстать сайты без использования HTML-таблиц.

Свойство display

- **none** - отключает отображение блока в окне браузера
- **block** – показывает блок (видимость)
- **inline** - значение определяет строчный блок
- **list-item** - элемент становится частью списка

Управление видимостью и переполнением блоков.

Свойство visibility

- **visible** - обычное состояние блока (по умолчанию), когда он нормально виден на экране

- **hidden** - блок становится прозрачным, т.е. невидимым.

Свойство overflow

- **visible**
- **hidden**
- **scroll**

```
<STYLE>
#small{
        border: 1px dotted #660033;
        padding: 0.5em;
        height: 60px;
        width: 150px;
        overflow: visible;
}
</STYLE>

<DIV ID="small">
Ширина данного блока 150 пикселей, а высота 60 пикселей. Проверим, удастся ли контролировать ситуацию с переполнением?
<DIV>
```

Схемы позиционирования

Схемы позиционирования

- Нормальный поток.
- Относительное позиционирование.
- Абсолютное позиционирование.
- Плавающая блоковая модель.

Схемы позиционирования.

Свойство POSITION

- **static** - блок будет считаться обычным, и позиционироваться в соответствии с правилами нормального потока.
- **relative** - относительное позиционирование (относительно нормального потока).
- **absolute** - абсолютное позиционирование.
- **fixed** - фиксированное позиционирование. Блок позиционируется абсолютно, а потом его положение фиксируется относительно области просмотра - такой блок не перемещается при прокрутке.

Относительное позиционирование

*Чтобы создать относительно позиционированный блок, необходимо записать свойство **position:relative**. В этом случае положение блока сначала будет вычислено относительно нормального потока, а затем блок будет смещен относительно этого места.*

- **left** - смещение левого края блока от левого края контейнера;
- **right** - смещение правого края блока относительно правого края контейнера;
- **top** - смещение верхнего края блока относительно верхнего края контейнера;
- **bottom** - смещение нижнего края блока относительно нижнего края контейнера.

```
.move {
        position:relative;
        left:20px;
        top:10px;
}
```

```
<DIV>первый блок </DIV>
<DIV class="move">второй блок </DIV>
<DIV>третий блок </DIV>
```

Абсолютное позиционирование

Если при относительном позиционировании блок не "вырывается" из нормального потока, а только смещается относительного нормального положения, то при абсолютном позиционировании все совершенно иначе: ***абсолютно позиционируемый блок*** *полностью "вырывается" из нормального потока.*

- **left** - смещение левого края блока относительно левого края контейнера;
- **right** - смещение правого края блока относительно правого края контейнера;
- **top** - смещение верхнего края блока относительно верхнего края контейнера;
- **bottom** - смещение нижнего края блока относительно нижнего края контейнера.

```
#abs {
        position: absolute;
        left:50px;
        top:100px;
}
```

```
<DIV>первый блок </DIV>
<DIV id="abs">второй блок </DIV>
<DIV>третий блок </DIV>
```

Фиксированные блоки

```
#fix{
        position: fixed;
        left:0px;
        top:0px;
        widht:100px;
}

#move {
        position: absolute;
        left:100px;
        top:0px;
}
```

```
<DIV id="fix">
Пункт 1<BR>
Пункт 2<BR>
Пункт 3<BR>
Пункт 4
</DIV>
<DIV id="move">
Основной текст страницы должен прокручиваться<BR>
Основной текст страницы должен прокручиваться<BR>
Основной текст страницы должен прокручиваться<BR>
.....
</DIV>
```

Плавающие блоки

- *Для того чтобы в **CSS реализовать "резиновую" верстку**, были придуманы плавающие блоки. Их нельзя позиционировать с точностью до пикселя, как, например, абсолютно позиционируемые. Они могут свободно перемещаться и "прижиматься" к краю своего контейнера.*
- *Подобным образом себя ведут рисунки в HTML, для которых задано выравнивание при помощи атрибута align.*

Свойства FLOAT

- **left** - структурный блок перемещается влево. Остальное содержимое документа будет выводиться вдоль правой стороны блока, начиная с самого верха.

- **right** - структурный блок перемещается вправо. Остальное содержимое документа выводится вдоль левой стороны блока, начиная с самого верха.

- **none** - блок не перемещается (значение по умолчанию), т.е. позиционируется согласно алгоритму, заданному свойством position. Если свойство position не задано, то предполагается нормальный поток.

Плавающие блоки

Пример

```
.comment {
        background: #FFC;
        border: 1px solid;
        padding: 5px;
        width: 150px;
        float: right;
}
```

```
<DIV class="comment">
Плавающий блок может "прилипать" к левой или
правой стороне контейнера, сторона задается
свойством <EM>float</EM>.
</DIV>
<P></P>
```

Плавающие блоки

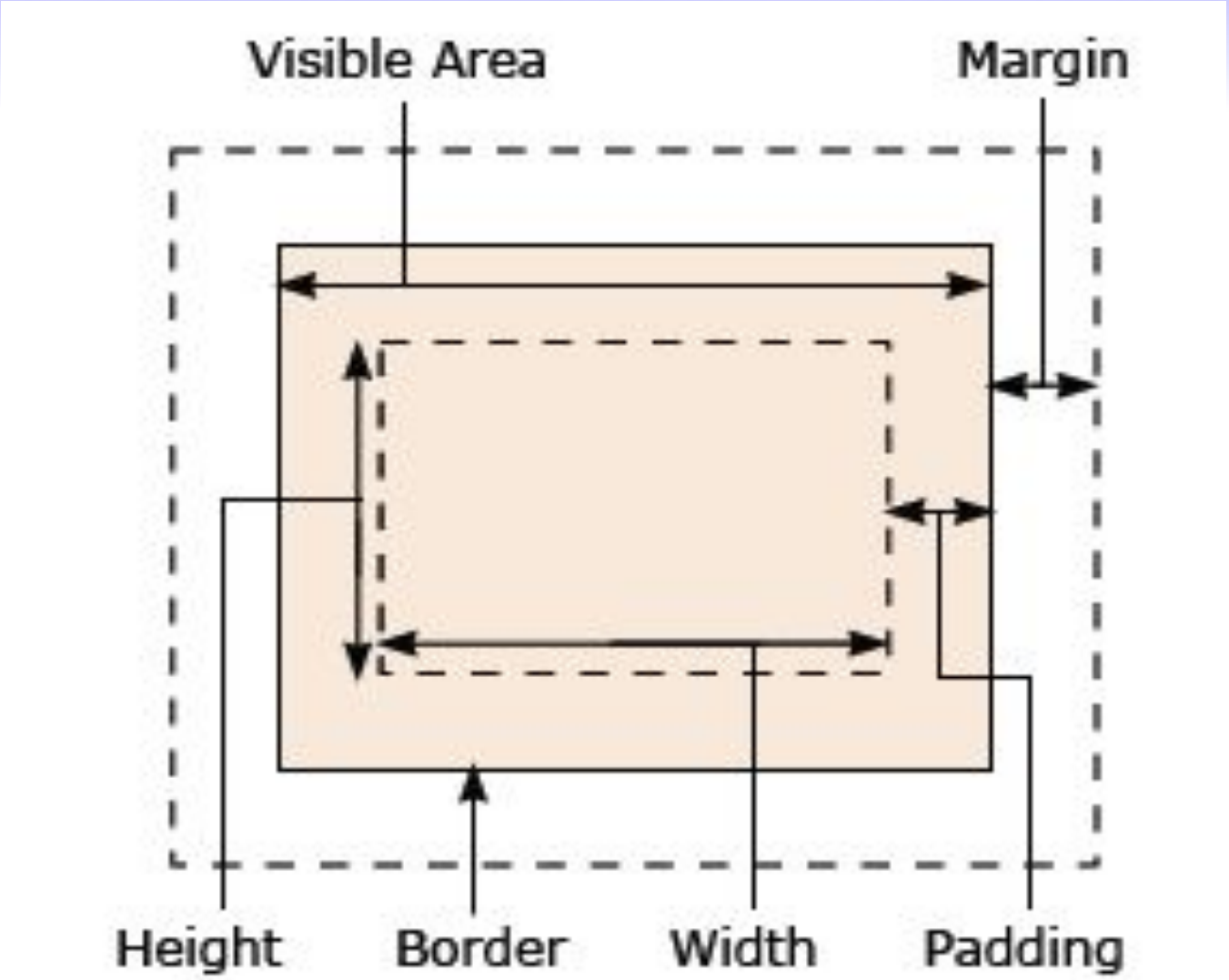
Свойства CLEAR

- **left** - блок должен размещаться ниже всех левосторонних плавающих блоков.
- **right** - блок должен размещаться ниже всех правосторонних плавающих блоков.
- **both** - блок должен размещаться ниже всех плавающих блоков.
- **none** - никаких ограничений на положение блока относительно перемещаемых объектов не накладывается.

```
.comment {
        background: #FFC;
        border: 1px solid;
        padding: 5px;
        width: 150px;
        float: right;
}
```

```
<DIV class="comment">
Первый плавающий блок должен прилипать к левому краю контейнера.
</DIV>
```

```
<DIV class="comment" style="clear:left">
Второй плавающий блок должен переместиться ниже первого блока. </DIV> <P>... некий фрагмент текста ...</P>
```


Visible Area
Margin
Height
Border
Width
Padding

Отступы

- **margin**
  Устанавливает величину отступа от каждого края элемента.

- **margin-top**
  Устанавливает величину отступа от верхнего края элемента.
- **margin-right**
  Устанавливает величину отступа от правого края элемента.
- **margin-left**
  Устанавливает величину отступа от левого края элемента.
- **margin-bottom**
  Устанавливает величину отступа от нижнего края элемента.

Поля

- **padding**
  Устанавливает значение полей вокруг содержимого элемента.

- **padding-top**
  Устанавливает значение поля от верхнего края содержимого элемента.
- **padding-right**
  Устанавливает значение поля от правого края содержимого элемента.
- **padding-left**
  Устанавливает значение поля от левого края содержимого элемента.
- **padding-bottom**
  Устанавливает значение поля от нижнего края содержимого элемента.

Рамки

- **border-color**
  Устанавливает цвет границы
- **border-width**
  Задает толщину границы
- **border-style**
  Устанавливает стиль границы

- **border**
  Позволяет одновременно установить толщину, стиль и цвет границы вокруг элемента.
  border: border-width border-style border-color;

Свойство border-style – стиль рамки

- dotted
- dashed
- solid
- double
- groove
- ridge
- inset
- outset

CSS

Cascading Style Sheets
Каскадные таблицы стилей
Язык описания внешнего вида документа

HTML - логическая структура
CSS - внешний вид

Отделяем внешний вид от структуры

CSS: как подключить к странице

1) Подключаем отдельный CSS файл

2) Пишем стили в HTML странице

3) Inline стили

CSS: отдельный файл

Может быть несколько css файлов.

```
<head>
  <link
        rel="stylesheet"
        type="text/css"
        href="css/global.css"/>
</head>
```

Стили в HTML странице

```
<html><head><title>…<title>
<style type="text/css">
p {color: green; font-weight: bold; font-family: Arial;}
</style>
</head>
<body>
<p>Этот текст имеет зеленый цвет и выводится жирным. Шрифт Arial. </p>
</body></html>
```

Этот текст имеет зеленый цвет и выводится жирным. Шрифт Arial.

Inline стили

<P **style**="**color:** green; **font-weight:** bold; **font-family:** Arial;">
Этот текст имеет зеленый цвет и выводится жирным. Шрифт Arial.
</P>

Этот текст имеет зеленый цвет и выводится жирным. Шрифт Arial.

CSS: правило

css файл состоит из перечня правил.

Каждое правило имеет вид:

```
селектор, селектор {
   свойство: значение;
   ...
}
```

CSS: как работает

HTML - иерархическая структура элементов.

Элемент:

```
<p class=”product discount” id=”p52”> … </p>
```

Вложенность:

```
<p>
    … <a><img></a> …
    <p> … </p>
</p>
```

CSS: виды селекторов

`* { }` - все элементы

`p { }` - выбор элемента

`.price { }` - выбор по классу

`#logo { }` - выбор по id элемента

`p a { }` - вложенные элементы

```
p a img { }
table tr td a {}
.product p.price b {}
```

CSS: свойства

font-family: Verdana;
color: red;
font-weight: bold;
font-size: 10pt;
white-space:nowrap;
text-decoration: line-through underline;
width: 100%;
text-align: left;

CSS: свойства

font-family: Verdana;
color: red;
font-weight: bold;
font-size: 10pt;
white-space:nowrap;
text-decoration: line-through underline;
width: 100%;
text-align: left;

CSS: примеры

```
* {
    font-family: Verdana;
}

table.product tr.header {
    font-weight: bold;
}
```

Преимущества стилей

- Разграничение кода и оформления
- Разное оформление для разных устройств
- Расширенные по сравнению с HTML способы оформления элементов
- Ускорение загрузки сайта
- Единое стилевое оформление множества документов
- Централизованное хранение

Шаблонизация

Способ отделения визуального представления от содержимого при создании динамических страниц.

Применяется для:

- Гибкого формирования представления,
- Повторного использования,
- Повышение эффективности сопровождения.

Уровень веб-компонент.

Ссылки

- Валидатор https://validator.w3.org/
- Проект https://gitlab.com/romanov73/ng-tracker
- Справочник https://webref.ru/html
- http://getbootstrap.com/docs/4.1/examples/
- http://getbootstrap.com/docs/4.1/components
- https://www.primefaces.org/
-