

КАК СТАТЬ АВТОРОМ



Да начнется битва Поехали в гик-трип по Календарю IT-...

Migratus

18 ноября 2013 в 11:03

Как сверстать веб-страницу. Часть 1

13 мин 1.2M

Веб-разработка*, CSS*, HTML*

Из песочницы

Уважаемый читатель, этой статьей я открываю цикл статей, посвященных вёрстке. В первой части будет описано, как это сделать с помощью стандартных средств на чистом HTML и CSS. В последующих частях рассмотрим как сделать тоже самое, но с помощью современных фреймворков и CMS.

Часть 1. Верстка стандартными средствами

Преимущество данной верстки состоит в том, что код получается более «чистым», а значит быстрее загружается и легче изменяется под специфические нужды. Недостаток такой верстки заключается в том, что она требует значительно больше времени, чем при использовании фреймворков.

Итак, давайте приступим. В качестве нашего подопытного мы возьмем бесплатный psd шаблон Согроте Blue от студии Pcklaboratory.

 whitesquare

Search GO

HOME ABOUT US SERVICES PARTNERS CUSTOMERS PROJECTS CAREERS CONTACT

ABOUT US

- LOREM IPSUM
- DONEC TINCIDUNT LAOREET
- VESTIBULUM ELIT
- ETIAM PHARETRA
- PHASELLUS PLACERAT
- CRAS ET NISI VITAE ODIO

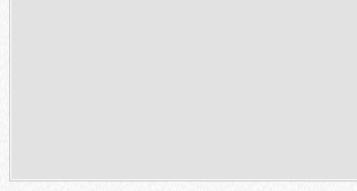
“QUISQUE IN ENIM VELIT, AT DIGNISSIM EST. NULLA UL CORPER, DOLOR AC PELLentesque PLACERAT, JUSTO TELLUS GRAVIDA ERAT, VEL PORTTITOR LIBERO ERAT.”

John Doe, Lorem Ipsum

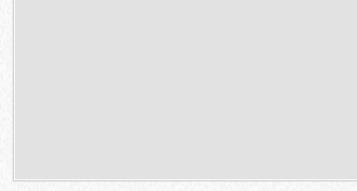
Lorem ipsum dolor sit amet, consectetur adipiscing elit. Aenean non neque ac sem accumsan rhoncus ut ut turpis. In hac habitasse platea dictumst. Proin eget nisi erat, et feugiat arcu. Duis semper porttitor lectus, ac pharetra erat imperdiet nec. Morbi interdum felis nulla. Aenean eros orci, pellentesque sed egestas vitae, auctor aliquam nisi. Nulla nec libero eget sem rutrum iaculis. Quisque in enim velit, at dignissim est. Nulla ullamcorper, dolor ac pellentesque placerat, justo tellus gravida erat, vel porttitor libero erat condimentum metus. Donec sodales aliquam orci id suscipit. Proin sed risus sit amet massa ultricies laoreet quis a erat. Aliquam et metus id erat vulputate egestas. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus.

Donec vel nisl nibh. Aenean quam tortor, tempus sit amet mattis dapibus, egestas tempor dui. Duis vestibulum imperdiet risus pretium pretium. Nunc vitae porta ligula. Vestibulum sit amet nulla quam. Aenean lacinia, ante vitae sodales sagittis, leo felis bibendum neque, mattis sagittis neque urna vel magna. Sed at sem vitae lorem blandit feugiat.

Donec vel orci purus, ut ornare orci. Aenean rutrum pellentesque quam. Quisque gravida adipiscing augue, eget commodo augue egestas varius. Integer volutpat, telus porta tincidunt sodales, lacus est tempus odio, fringilla blandit tortor lectus ut sem. Pellentesque nec sem lacus, sit amet consequat neque. Etiam varius urna quis arcu cursus in consectetur duis tincidunt. Quisque arcu orci, lacinia eget pretium vel, iaculis pellentesque nibh. Etiam cursus lacus eget neque viverra vestibulum. Aliquam erat volutpat. Duis pulvinar tellus, ut urna facilisis mollis. Maecenas ac pharetra dui. Pellentesque neque ante, luctus eget congue augue, rhoncus vel mauris. Duis nisi magna, aliquet a convallis non, venenatis at nisl. Nunc at quam eu magna malesuada dignissim. Duis bibendum iaculis felis, eu venenatis risus sodales non. In ligula mi, faucibus eu tristique sed, vulputate rutrum dolor.



John Doe
ceo



Saundra Pittsley
team leader



Julio Simser
senior developer



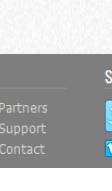
Margery Venuti
senior developer



Fernando Tondrea
developer



Ericka Nobriga
art director



Cody Rousselle
senior ui designer



Erik Wollman
senior ui designer

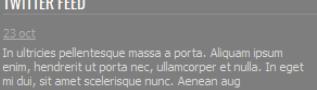


Dona Shoff
ux designer



Darryl Brunton
ui designer

OUR TEAM



23 oct
In ultricies pellentesque massa a porta. Aliquam ipsum enim, hendrerit ut porta nec, ullamcorper et nulla. In eget mi dui, sit amet scelerisque nunc. Aenean aug

SITEMAP

[Home](#) [About](#) [Services](#) [Partners](#) [Support](#) [Contact](#)

SOCIAL NETWORKS



 whitesquare
Copyright © 2012 Whitesquare. A [pickab creation](#)

Структура файлов

Первым шагом давайте создадим простую структуру файлов для наших файлов.

- Создаем папку с названием нашего проекта, например `Whitesquare`.
- В ней создаем пустой файл `index.html`.
- В папке проекта создаем папку `css` с пустым файлом `styles.css`.

- В папке проекта создаем пустую папку `images`.



Предварительный осмотр

После создания структуры файлов открываем `psd` файл в Photoshop. Важно внимательно осмотреть шаблон и оценить его. Нам нужно понять следующие вещи:

- Как будут нарезаться изображения?
- Какими будут основные стили?
- Какой макет у нас получится?

Только после того, как вы мысленно себе ответите на эти вопросы, можно переходить к нарезке изображений и написанию кода. Давайте рассмотрим эти вопросы по-порядку.

Общие изображения

На данном этапе нужно нарезать и сохранить только общие изображения, которые будут на всех страницах сайта и не относятся к контенту. В нашем случае это будет светло-серый фон страницы, фон заголовка, пустое изображение, два логотипа и кнопки социальных сетей.

Сохраним логотипы следующим образом:

`/images/logo.png`

`/images/footer-logo.png`

В качестве пустых картинок из макета будем использовать однопиксельное серое изображение, которое будем растягивать по необходимости

`/images/ sample.png`

Повторяющиеся фоновые изображения необходимо вырезать минимальным кусочком достаточным для образования полного изображения повторением по вертикали и горизонтали.

`/images/bg.png`

/images/h1-bg.png

Иконки социальных сетей с одинаковыми размерами удобно сохранить в один файл и использовать как спрайты для более быстрой загрузки. Для этого можно склеить картинки вручную в Photoshop, а можно сначала нарезать по одной, а затем склеить с помощью специально сервиса, например <http://ru.spritegen.website-performance.org>. В итоге получится два файла:

/images/social.png

/images/social-small.png

Общее правило при именовании изображений заключается в том, что мелкие и простые картинки, такие, как иконки, логотипы и т.д. сохраняются в формате png, а фотографии в формате jpg.

Основные стили

И только теперь можно начинать писать код. Но начнем мы это делать не с привычного HTML, а с переноса правил в CSS.

На данном этапе желательно перенести все визуальные стили из дизайна в CSS, которые будут применяться по умолчанию для каждого тега.

Основной цвет фона примерно соответствует цвету #f8f8f8. Он будет показан в случае, если фоновая картинка не загрузится. Наверху страницы находится серая дизайннерская полоска. Применим ее через свойство border для body.

Основным шрифтом является тот шрифт, которым написан текст в области контента. Чтобы узнать его стили нужно выделить его в Photoshop'е и посмотреть свойства шрифта. В данном случае это Tahoma 12px с цветом #8f8f8f. Так же в этом макете параграфы имеют увеличенные отступы.

Прописываем все эти стили в styles.css:

```
body {  
    color: #8f8f8f;  
    font: 12px Tahoma, sans-serif;  
    background-color: #f8f8f8;  
    border-top: 5px solid #7e7e7e;  
    margin: 0;  
}  
  
input[type="text"] {  
    background-color: #f3f3f3;
```

```
border: 1px solid #e7e7e7;
height: 30px;
color: #b2b2b2;
padding: 0 10px;
vertical-align: top;
}

button {
    color: #fff;
background-color: #29c5e6;
border: none;
height: 32px;
font-family: 'Oswald', sans-serif;
}

p {
    margin: 20px 0;
}
```

В дальнейшем мы все стили будем писать в этот же файл, поэтому будем называть его просто «стили».

Каркас HTML

И вот, наконец, мы можем попрактиковаться в написании HTML кода. Запишем в `index.html` следующее:

```
<!doctype html>
<html>
<head>
    <meta http-equiv="Content-type" content="text/html; charset=utf-8">
    <meta http-equiv="X-UA-Compatible" content="IE=Edge">
    <title>Whitesquare</title>
    <link rel="stylesheet" href="css/styles.css" type="text/css">
    <link rel="stylesheet" href="http://fonts.googleapis.com/css?family=Oswald:400,700">

    <!--[if lt IE 9]>
    <script src="http://html5shiv.googlecode.com/svn/trunk/html5.js"></script>
    <![endif]-->
</head>
<body>
</body>
</html>
```

Здесь мы указываем, что используем разметку HTML5, кодировку utf-8, страница называется Whitesquare. Также подключаем наш файл стилей и внешний файл со стилями шрифтов.

В последнем блоке в секции head мы подключаем специальный скрипт, который позволяет поддерживать Html5 теги в браузерах Internet Explorer меньше 9 версии. Мета-тег X-UA-Compatible сообщает, что в случае использования браузера Internet Explorer, он должен отразить сайт самым современным способом.

Весь html код в дальнейшем будет относиться к этому же файлу, поэтому специально указывать куда прописывать html код автор больше не будет.

Макет

В данном случае, мы видим, что сайт состоит из двух колонок: основного контента и сайдбара. Над ними находится шапка (header), в которой располагаются три горизонтальных блока: логотип с поиском, меню и название страницы. В самом низу под колонками располагается серый горизонтальный блок футера (footer).

WRAPPER

whitesquare

SEARCH GO

HEADER

ABOUT US

HOME SERVICES PARTNER PROJECTS CAREERS CONTACT

ABOUT US

- LOREM IPSUM
- DONEC TINCIDUNT LAOREET
- VESTIBULUM ELIT
- ETIAM PHARETRA
- PHASELLUS PLACERAT
- CRAS ET NISI VITAE ODIO

DUR OFFICES

SIDE BAR

“*QUISQUE IN ENIM VELIT, AT DIGNISSIM EST. NULLA UL CORPER, DOLOR AC PELLentesque PLACERAT, JUSTO TELLUS GRAVIDA ERAT, VEL PORTTITOR LIBERO ERAT.*”

John Doe, Lorem ipsum

lorem ipsum dolor sit amet, consectetur adipiscing elit. Aenean non neque ac sem accumsan rhoncus ut turpis. In hac habitasse platea dictumst. Proin eget nisi erat, et feugiat arcu. Duis semper porttitor lectus, ac pharetra erat imperdiet nec morbi interdum felis nulla. Aenean eros orci, pellentesque sed egestas vitae, auctor aliquam nisi. Nulla nec libero eget sem utrum iaculis. Quisque in enim velit, at dignissim est. Nulla ullamcorper, dolor ac pellentesque placerat, justo tellus gravida erat, vel porttitor libero erat condimentum metus. Donec sodales aliquam orci id suscipit. Proin sed risus sit amet massa ultrices laoreet quis a erat. Aliquam et metus id erat vulputate egestas. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus.

Donec vel nisl nibh. Aenean quam tortor, tempus sit amet mattis dapibus, egestas tempor dui. Duis vestibulum imperdiet sus pretium pretium. Nunc vitae porta ligula. Vestibulum sit amet nulla quam. Aenean lacinia, ante vitae sodales sagittis, lectus bibendum neque, mattis sagittis neque urna vel magna. Sed at sem vitae lorem blandit feugiat.

Donec vel orci purus, ut ornare orci. Aenean rutrum pellentesque quam. Quisque gravida adipiscing augue, eget commodo augue egestas varius. Integer volutpat ante. In hac habitasse platea dictumst. Proin eget nisi erat, et feugiat arcu. Duis nec sem arcus cursus in consectetur dui tincidunt. Quisque arcu orci, lacinia eget orci. Ut varius pellentesque nibh. Etiam cursus lacus eget neque viverra vestibulum. Aliquam erat volutpat. Duis pulvinar tellus ut urna facilis mollis. Maecenas ac pharetra dui. Pellentesque neque ante, luctus eget congue eget, rhoncus vel mauris. Duis nisi magna, aliquet a convallis non, venenatis at nisl. Nunc at quam eu magna malesuada dignissim. Duis bibendum iaculis felis, eu venenatis risus sodales non. In ligula mi, faucibus eu tristique sed, vulputate rutrum dolor.

OUR TEAM

John Doe CEO	Sandra Pittsley team leader	Julio Simser senior developer	Margery Venuti senior developer	Fernando Tondre developer
Ericka Nobriga art director	Cody Rousselle senior ui designer	Erik Wollman senior ui designer	Dona Shoff ux designer	Darryl Brunton ui designer

TWITTER FEED

23 oct

In ultricies pellentesque massa a porta. Aliquam ipsum enim, hendrerit ut porta nec, ullamcorper et nulla. In egestas mi dui, sit amet scelerisque nunc. Aenean aug.

SITEMAP

Home About Services Partners Support Contact

SOCIAL NETWORKS

FOOTER

whitesquare

Copyright © 2012 Whitesquare. A [pcklab](#) creation

Опишем это в теге body:

```
<body>
<div id="wrapper">
  <header></header>
  <nav></nav>
  <div id="heading"></div>
  <aside></aside>
```

```
<section></section>
</div>
<footer></footer>
</body>
```

Wrapper используется для объединения блоков и их выравнивания по центру страницы.

Затем укажем стили блоков:

```
#wrapper {
    max-width: 960px;
    margin: auto;
}

header {
    padding: 20px 0;
}
```

Логотип

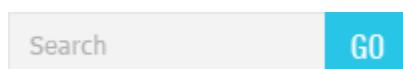


Вставляем логотип в тег header:

```
<header>
    <a href="/"><img src="" alt="Whitesquare logo"></a>
</header>
```

Дополнительных стилей не требуется.

Поиск



Вставляем форму поиска в тег header:

```
<header>
...
<form name="search" action="#" method="get">
    <input type="text" name="q" placeholder="Search"><button type="submit">GO</button>
</form>
</header>
```

И стили выравнивания по правому краю для нее:

```
form[name="search"] {
    float: right;
}
```

Меню

Для отображения меню необходимо создать список со ссылками внутри тега nav:

```
<nav>
    <ul class="top-menu">
        <li><a href="/home/">HOME</a></li>
        <li class="active">ABOUT US</li>
        <li><a href="/services/">SERVICES</a></li>
        <li><a href="/partners/">PARTNERS</a></li>
        <li><a href="/customers/">CUSTOMERS</a></li>
        <li><a href="/projects/">PROJECTS</a></li>
        <li><a href="/careers/">CAREERS</a></li>
        <li><a href="/contact/">CONTACT</a></li>
    </ul>
</nav>
```

CSS стили для него будут следующие:

```
nav a {  
    text-decoration: none;  
}  
  
nav ul {  
    margin: 0;  
    padding: 0;  
}  
  
nav li {  
    list-style-position: inside;  
    font: 14px 'Oswald', sans-serif;  
    padding: 10px;  
}  
  
.top-menu li {  
    display: inline-block;  
    padding: 10px 30px;  
    margin: 0;  
}  
  
.top-menu li.active {  
    background: #29c5e6;  
    color: #fff;  
}  
  
.top-menu a {  
    color: #b2b2b2;  
}
```

Здесь мы указали, что для всех навигаций ссылки не будут иметь подчеркивания, убрали стандартные отступы для элементов списка, отобразили список горизонтально и указали нужные цвета и шрифт.

Заголовок страницы

ABOUT US

Заголовок страницы помещается в div с идентификатором heading

```
<div id="heading">  
    <h1>ABOUT US</h1>
```

</div>

Заголовок имеет следующие стили:

```
#heading {  
    background: transparent url(..../images/h1-bg.png);  
    margin: 30px 0;  
    padding-left: 20px;  
}  
  
h1 {  
    display: inline-block;  
    color: #7e7e7e;  
    font: 40px/40px 'Oswald', sans-serif;  
    background: url(..../images/bg.png);  
    margin: 0;  
    padding: 0 10px;  
}
```

Рисуем серую полоску фоном на div'e, и в нее вкладываем инлайновый h1 с нужным шрифтом и фоном цвета страницы, чтобы создалось впечатление прозрачного фона для h1.

Колонки

Для того, чтобы создать колонки страницы нужно прописать следующие стили:

```
aside {  
    float: left;  
    width: 250px;  
}  
  
section {  
    margin-left: 280px;  
    padding-bottom: 50px;  
}
```

Здесь мы задали фиксированную ширину 250 пикселей для сайдбара, прибили его к левому краю и отодвинули колонку с контентом вправо на 280 пикселей от левого края. Также

добавили отступ у контента снизу.

Подменю



Подменю создаем аналогично главному меню. Для этого в теге aside прописываем следующее:

```
<aside>
<nav>
  <ul class="aside-menu">
    <li class="active">LOREM IPSUM</li>
    <li><a href="/donec/">DONEC TINCIDUNT LAOREET</a></li>
    <li><a href="/vestibulum/">VESTIBULUM ELIT</a></li>
    <li><a href="/etiam/">ETIAM PHARETRA</a></li>
    <li><a href="/phasellus/">PHASELLUS PLACERAT</a></li>
    <li><a href="/cras/">CRAS ET NISI VITAE ODIO</a></li>
  </ul>
</nav>
</aside>
```

И применяем к подменю следующие стили:

```
.aside-menu li {
  font-weight: 300;
  list-style-type: square;
  border-top: 1px solid #e7e7e7;
}

.aside-menu li:first-child {
  border: none;
```

```

}

.aside-menu li.active {
    color: #29c5e6;
}

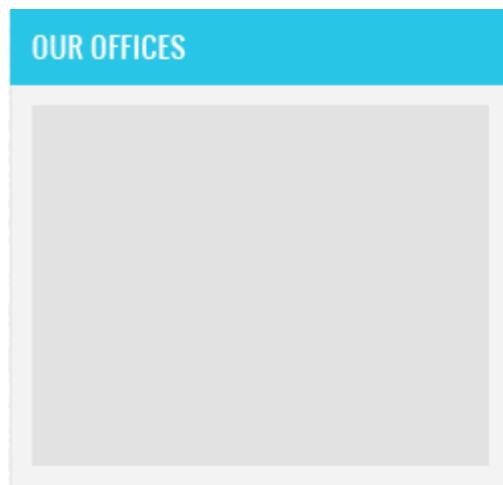
.aside-menu a {
    color: #8f8f8f;
}

```

Для подменю применяется более тонкий шрифт и квадратные маркеры. Для отображения разделителей рисуем верхнюю границу у каждого элемента списка, кроме первого.

Контент сайдбара

В контенте сайдбара помимо подменю в макете располагается также изображение с расположением офисов.



В html это выглядит так:

```

<h2>OUR OFFICES</h2>
<p></p>

```

В стилях укажем шрифты, цвета и отступы:

```

aside > h2 {
    background: #29c5e6;
    font: 14px 'Oswald', sans-serif;
}

```

```
color: #fff;
padding: 10px;
margin: 30px 0 0 0;
}

aside > p {
background: #f3f3f3;
border: 1px solid #e7e7e7;
padding: 10px;
margin: 0;
}
```

Данные стили применяются только к заголовкам и параграфам, лежащим непосредственно внутри сайдбара, но не глубже.

Цитата

Вёрстку контента начнём с добавления цитаты.



Добавим код цитаты в раздел section

```
<section>
<blockquote>
  <p>
    "QUISQUE IN ENIM VELIT, AT DIGNISSIM EST. NULLA UL CORPER, DOLOR AC PELL
  </p>
  <cite>John Doe, Lorem Ipsum</cite>
</blockquote>
</section>
```

И применим для него стили:

```
blockquote {  
    margin: 0;  
    background: #29c5e6;  
    padding: 10px 20px;  
    font-family: 'Oswald', sans-serif;  
    font-weight: 300;  
}  
  
blockquote p {  
    color: #fff;  
    font-style: italic;  
    font-size: 33px;  
    margin: 0;  
}  
  
blockquote cite {  
    display: block;  
    font-size: 20px;  
    font-style: normal;  
    color: #1d8ea6;  
    margin: 0;  
    text-align: right;  
}
```

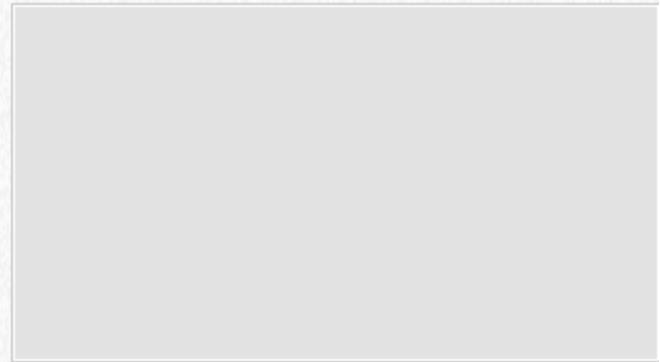
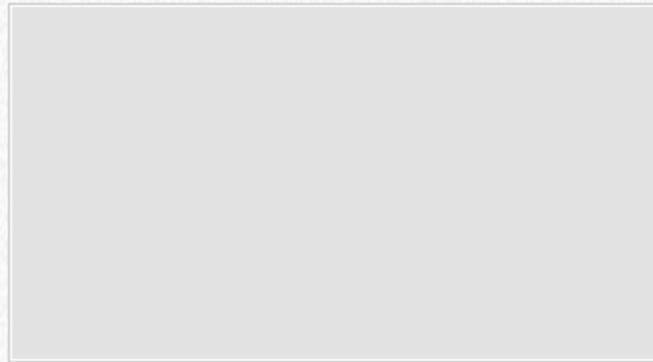
Здесь нет ничего нового, так же – шрифты, фоны и отступы.

Контент

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Aenean non neque ac sem accumsan rhoncus ut ut turpis. In hac habitasse platea dictumst. Proin eget nisi erat, et feugiat arcu. Duis semper porttitor lectus, ac pharetra erat imperdiet nec. Morbi interdum felis nulla. Aenean eros orci, pellentesque sed egestas vitae, auctor aliquam nisi. Nulla nec libero eget sem rutrum iaculis. Quisque in enim velit, at dignissim est. Nulla ullamcorper, dolor ac pellentesque placerat, justo tellus gravida erat, vel porttitor libero erat condimentum metus. Donec sodales aliquam orci id suscipit. Proin sed risus sit amet massa ultrices laoreet quis a erat. Aliquam et metus id erat vulputate egestas. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus.

Donec vel nisl nibh. Aenean quam tortor, tempus sit amet mattis dapibus, egestas tempor dui. Duis vestibulum imperdiet risus pretium pretium. Nunc vitae porta ligula. Vestibulum sit amet nulla quam. Aenean lacinia, ante vitae sodales sagittis, leo felis bibendum neque, mattis sagittis neque urna vel magna. Sed at sem vitae lorem blandit feugiat.

Donec vel orci purus, ut ornare orci. Aenean rutrum pellentesque quam. Quisque gravida adipiscing augue, eget commodo augue egestas varius. Integer volutpat, tellus porta tincidunt sodales, lacus est tempus odio, fringilla blandit tortor lectus ut sem. Pellentesque nec sem lacus, sit amet consequat neque. Etiam varius urna quis arcu cursus in consectetur dui tincidunt. Quisque arcu orci, lacinia eget pretium vel, iaculis pellentesque nibh. Etiam cursus lacus eget neque viverra vestibulum. Aliquam erat volutpat. Duis pulvinar tellus ut urna facilisis mollis. Maecenas ac pharetra dui. Pellentesque neque ante, luctus eget congue eget, rhoncus vel mauris. Duis nisi magna, aliquet a convallis non, venenatis at nisl. Nunc at quam eu magna malesuada dignissim. Duis bibendum iaculis felis, eu venenatis risus sodales non. In ligula mi, faucibus eu tristique sed, vulputate rutrum dolor.



Все стили для текста контента мы уже добавили. Поэтому остается добавить только три параграфа с самим текстом после

```
<p>Lorem ipsum dolor sit amet...</p>
<p>Donec vel nisl nibh...</p>
<p>Donec vel nisl nibh...</p>
```

Следующим шагом нужно добавить два изображения, которые находятся в конце текста контента. Делается это с помощью тега :

```
<figure>
  
</figure>
<figure>
  
</figure>
```

, которому зададим следующие стили:

```
figure {
  display: inline-block;
  margin: 0;
  font-family: 'Oswald', sans-serif;
  font-weight: 300;
}

figure img {
  display: block;
  border: 1px solid #fff;
  outline: 1px solid #c9c9c9;
}

figure figcaption {
  font-size: 16px;
  font-weight: 300;
  margin-top: 5px;
```

+85

2638



110

```
figure figcaption span {
  display: block;
  font-size: 14px;
  color: #29c5e6;
}

section > figure + figure {
  margin-left: 28px;
}
```

Здесь мы убрали стандартные отступы у `figure`, отобразили его как инлайновый блок и применили нужный шрифт. Изображение отображаем как

блочного элемента с белой рамкой. Вторую серую рамку можно сделать через CSS-свойство `outline`. Самое интересное находится в последнем правиле, которое задает левый отступ у всех `figure` кроме первого внутри тега `section`.

Блок «Our team»



При верстке этого блока добавим сначала заголовок:

```
<h2>OUR TEAM</h2>
```

Со стилем:

```
section > h2 {
    background: #29c5e6;
    font: 30px 'Oswald', sans-serif;
    font-weight: 300;
    color: #fff;
    padding: 0 10px;
    margin: 30px 0 0 0;
}
```

А затем два блока-строки с карточками сотрудников

```
<div class="team-row">
  <figure>
    
    <figcaption>John Doe<span>ceo</span></figcaption>
  </figure>
  <figure>
    
    <figcaption>Saundra Pittsley<span>team leader</span></figcaption>
  </figure>
  ...
</div>
<div class="team-row">
  <figure>
    
    <figcaption>Ericka Nobriga<span>art director</span></figcaption>
  </figure>
  <figure>
    
    <figcaption>Cody Rousselle<span>senior ui designer</span></figcaption>
  </figure>
  ...
</div>
```

Таким образом, карточка (figure) состоит из фотографии (img), подписи (figcaption) с именем сотрудника и его должностью (div). Карточки будут иметь следующие стили:

```
figure figcaption {
  font-size: 16px;
  font-weight: 300;
  margin-top: 5px;
}

figure div {
  font-size: 14px;
  color: #29c5e6;
}
```

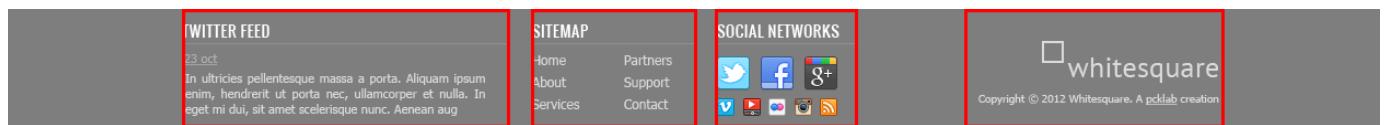
```
.team-row figure {
    margin-top: 20px;
}

.team-row figure + figure {
    margin-left: 43px;
}
```

Здесь мы задали шрифт и отступ для подписи, размер и цвет должности, добавили верхний отступ для карточек и указали, что для всех карточек в строке, кроме первой, должен быть отступ слева.

Футер

Футер состоит из четырёх больших блоков: ленты Твиттера, карты сайта, социальных ссылок и логотипа с копирайтом.



Для начала создадим контейнер футера с этими блоками:

```
<footer>
    <div id="footer">
        <div id="twitter"></div>
        <div id="sitemap"></div>
        <div id="social"></div>
        <div id="footer-logo"></div>
    </div>
</footer>
```

И применим к нему оформление:

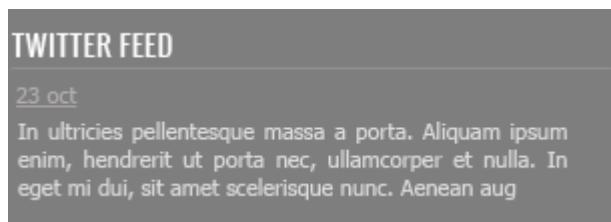
```
footer {
    background: #7e7e7e;
    color: #dbdbdb;
    font-size: 11px;
}
```

```
#footer {
    max-width: 960px;
    margin: auto;
    padding: 10px 0;
    height: 90px;
}
```

Контейнер с id="footer" находится внутри тега footer, это даёт нам возможность через тег footer задать серую область по всей ширине экрана, а через внутренний div отцентрировать с максимальной шириной 960 пикселей. Также этот div задает обоим блокам высоту 90 пикселей.

Лента Твиттера

Верстаем содержимое ленты Твиттера:



```
<div id="twitter">
    <h3>TWITTER FEED</h3>
    <time datetime="2012-10-23"><a href="#">23 oct</a></time>
    <p>
        In ultricies pellentesque massa a porta. Aliquam ipsum enim, hendrerit ut poi
    </p>
</div>
```

Стили:

```
footer h3 {
    font: 14px 'Oswald', sans-serif;
    color: #fff;
    border-bottom: 1px solid #919191;
    margin: 0 0 10px 0;
```

```
}

#twitter time a {
    color: #b4aeae;
}

footer p {
    margin: 5px 0;
}

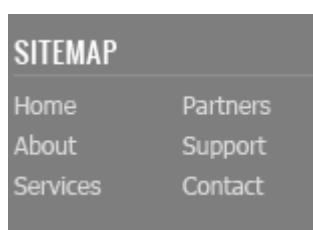
#twitter {
    float: left;
    width: 300px;
}

#twitter p {
    padding-right: 15px;
}
```

Из интересных моментов здесь следующее: подчеркивание у заголовка мы сделали через нижнюю границу, а сам блок твиттера, как и последующие блоки выровняем по левому краю и задаём ширину.

Карта сайта

Карта сайта представляет собой два блока со ссылками:



```
<div id="sitemap">
    <h3>SITEMAP</h3>
    <div>
        <a href="/home/">Home</a>
        <a href="/about/">About</a>
        <a href="/services/">Services</a>
    </div>
    <div>
        <a href="/partners/">Partners</a>
```

```
<a href="/customers/">Support</a>
<a href="/contact/">Contact</a>
</div>
</div>
```

Ссылкам задаем цвет и оставляем подчёркивание только для наведённых. Колонки со ссылками делаем через инлайновые блоки и затем свойством #sitemap div + div отодвигаем вторую колонку от первой.

```
footer a {
    color: #dbdbdb;
}

#sitemap {
    width: 150px;
    float: left;
    margin-left: 20px;
    padding-right: 15px;
}

#sitemap div {
    display: inline-block;
}

#sitemap div + div {
    margin-left: 40px;
}

#sitemap a {
    display: block;
    text-decoration: none;
    font-size: 12px;
    margin-bottom: 5px;
}

#sitemap a:hover {
    text-decoration: underline;
}
```

Социальные ссылки



Вставляем набор ссылок в контейнер

```
<div id="social">
    <h3>SOCIAL NETWORKS</h3>
    <a href="http://twitter.com/" class="social-icon twitter"></a>
    <a href="http://facebook.com/" class="social-icon facebook"></a>
    <a href="http://plus.google.com/" class="social-icon google-plus"></a>
    <a href="http://vimeo.com/" class="social-icon-small vimeo"></a>
    <a href="http://youtube.com/" class="social-icon-small youtube"></a>
    <a href="http://flickr.com/" class="social-icon-small flickr"></a>
    <a href="http://instagram.com/" class="social-icon-small instagram"></a>
    <a href="/rss/" class="social-icon-small rss"></a>
</div>
```

И стилизуем их:

```
#social {
    float: left;
    margin-left: 20px;
    width: 130px;
}

.social-icon {
    width: 30px;
    height: 30px;
    background: url(../images/social.png) no-repeat;
    display: inline-block;
    margin-right: 10px;
}

.social-icon-small {
```

```
width: 16px;
height: 16px;
background: url(../images/social-small.png) no-repeat;
display: inline-block;
margin: 5px 6px 0 0;
}

.twitter {
background-position: 0 0;
}

.facebook {
background-position: -30px 0;
}

.google-plus {
background-position: -60px 0;
}

.vimeo {
background-position: 0 0;
}

.youtube {
background-position: -16px 0;
}

.flickr {
background-position: -32px 0;
}

.instagram {
background-position: -48px 0;
}

.rss {
background-position: -64px 0;
}
```

Здесь мы применили технику спрайтов – когда один файл с изображением применяется для разных картинок. Все ссылки разделились на большие иконки (.social-icon) и маленькие (.social-icon-small). Мы задали этим классом отображение в виде инлайнового блока с фиксированными

размерами и одинаковым фоном. А затем с помощью CSS сдвинули этот фон так, чтобы на каждой ссылке отобразилось соответствующее изображение.

Копирайт



Блок с копирайтом и логотипом – это картинка со ссылкой и параграф с текстом под ним.

```
<div id="footer-logo">
  <a href="/"><img src="" alt="Whitesquare logo"></a>
  <p>Copyright © 2012 Whitesquare. A <a href="http://pcklab.com">pcklab</a> cr</p>
</div>
```

Стили делают аналогично предыдущим блокам с той лишь разницей, что блок прибивается к правому краю и выравнивание внутри него так же по правому краю:

```
#footer-logo {
  float: right;
  margin-top: 20px;
  font-size: 10px;
  text-align: right;
}
```

На этом наши работы закончены. Готовый проект можно скачать [здесь](#).

Теги: html-верстка, css, tutorial

Хабы: Веб-разработка, CSS, HTML

Редакторский дайджест

Присыпаем лучшие статьи раз в месяц



Электропочта

忍 54 0
Карма Рейтинг

Михаил @Mirantus

Пользователь

Реклама



practicum.yandex.ru РЕКЛАМА · 16+ Я

Курс «Специалист по Data Science» от Яндекса

Обучаем специалистов в Data Science с нуля. 20 часов практики – бесплатно.

Комментарии 110



● aen 18 ноя 2013 в 11:44

А почему сразу не писать на less/sass?

◆ -17 Ответить



● 忍 Mirantus 18 ноя 2013 в 12:26

В данной статье была цель описать процесс верстки страницы максимально нативными средствами. В следующей части я опишу как сделать то же самое, но с помощью Twitter Bootstrap, который как раз использует LESS.

◆ +15 Ответить



● GruZZ 18 ноя 2013 в 16:51

А потом можно сделать nightmare-вариант с bem-tools, grunt, автогенерацией спрайтов и т.д. и т.п.

◆ +8 Ответить



● str4n9er 18 ноя 2013 в 12:26

Спасибо за очень подробное описание!

+2 Ответить



xplim
18 ноя 2013 в 12:49

Ваш код получается очень специфичный. Такой код совершенно спокойно подходит для написания страниц, которые не нужно будет затем менять, но не для проектов, которые постоянно развиваются.

Использование стилей для `id` элементов не есть хорошо.

При последующей поддержке такого сайта Вы пойдете в `inline` стили элементов.

Необходимо принять, что стили для `id` – это плохо.

Для стилей лучше использовать `class`. `id` – использовать для `js` кода.

Главная задача хорошо сверстанной страницы – возможность перемещения блоков в любое место. Простой пример – у вас есть блок новости.

Теперь в блок новости нужно вставить не только сами новости, но и скажем, блок погоды. В данном случае, если у вас используется каскадная верстка (`.class div div` и иже с ними), то Вам придется переписывать верстку для погоды и новостей. В большом проекте – это большое время и затраты. (сменили в одном месте – вполне может упасть в другом. Опять придется «выплывать» за счет `inline` стилей. (либо `!important`) Проект станет менее понятным.

При верстке проектов лучше подходить с точки зрения какой-либо из методологий верстки – (Для российского сегмента интернета самой популярной методологией является БЭМ, разработанный в недрах яндекса)

+27 Ответить



DarthSim
18 ноя 2013 в 09:00

`id` – использовать для `js` кода

Я бы сказал, что использование `id` для `js` тоже зло. Для `js` лучше пользоваться ролями или `.js-*` классами. По той же причине: не ровен час как придется то же поведение применить к нескольким блокам.

+3 Ответить



xplim
18 ноя 2013 в 09:04

Это вопрос уже ближе к стилю кода. Те же классы могут повторяться и т.д.

Прелест использования `id` для `js` – их уникальность, соответственно – поиск по DOM модели будет быстрее (при найденном элементе – поиск далее продолжать не нужно)

 +9

Ответить



...

**DarthSim**

18 ноя 2013 в 09:12

О чём, собственно, и речь. Редко когда нужно задать поведение некоего уникального элемента. Обычно это набор элементов с одинаковым поведением. Или одному элементу нужно задать несколько не уникальных действий. Соответственно, `id` становится неприменимым в данном случае. А биндить события на `id` для одних элементов, а для других – на классы, имхо, моветон, лучше уж выбрать один стиль. Искать бинды опять же проще, если знаешь, к какому селектору они привязаны.



Ответить



...

**xplim**

18 ноя 2013 в 09:13

Здесь с Вами согласен.



Ответить



...

**navstyachka**

18 ноя 2013 в 14:43

Для js я использую классы, написанные в camelcase, в то время как остальное пишу в системе БЭМ, тогда быстро можно разобраться, даже если проект давешний.



Ответить



...

**lastwish**

18 ноя 2013 в 15:00

А почему не data-атрибуты для js?



Ответить



...

**navstyachka**

18 ноя 2013 в 15:06

Когда-то пришла к такому стилю написания, теперь использую всегда. Не знаю, насколько для вас будет весомым следующий аргумент, но когда я пишу на haml, мне очень удобно использовать именно camelcase.



Ответить



...

**Mirantus**

18 ноя 2013 в 10:20

Большое спасибо за внимательное прочтение статьи и ваш комментарий. Он будет очень полезен новичкам. Действительно, использовать идентификаторы в css – не самая лучшая идея, особенно когда идет речь о сложных проектах.

В данном случае я их использовал по двум причинам.

Первая – это конечно же потому, что описывал процесс верстки одной конкретной

страницы, а не сайта, что отражено в заголовке.

А второе – это вопрос удобства чтения CSS. Если в коде я встречаю правило `.heading {...}`, то я не могу быть точно уверен к какой части страницы оно применится. Может быть это заголовок одного блока или другого. При верстке данной страницы я использовал идентификаторы только для глобальных или уникальных блоков. Поэтому, читая правило `#heading {...}`, я понимаю, что речь идет о самом главном заголовке на странице и он всегда один.

Во всём остальном – вы абсолютно правы.

За рекомендацию использовать БЭМ – отдельное спасибо. Вот ссылка на неё для всех остальных.

 +1 Ответить

 ...

 **xplim**
18 ноя 2013 в 10:26

▲

Если Вы в коде встречаете `.heading` без дополнительной информации – главное, что вы должны полагать – это шапка сайта.

Если же `heading` «тыкается» в каждый блок – это пример плохого сайта.

Главная проблема верстки, на мой скромный взгляд, когда разработчик говорит себе – тут сайт маленький, одна страница… сделаю-ка я его каскадом и `id`… А плохо это потому, что:

- 1) Может потребоваться изменить сайт (даже статический, да)
- 2) Эта болезнь может передаться новичкам
- 3) Разработчик может решить – «да я писал на `id` css, все работает, сделаю-ка я и в большом проекте так же.»

 +3 Ответить

 ...

 **Makaveli**
18 ноя 2013 в 15:37

▲

Для российского сегмента интернета самой популярной методологией является БЭМ, разработанный в недрах яндекса

wat? это по каким это данным/критериям? По-моему Яндекс != «Российский сегмент интернета»

 +1 Ответить

 ...

 **Myuzik**
18 ноя 2013 в 09:07

Почему вы опираетесь на тэги, ID и каскадные селекторы в вашем css? Если хотите кого-то чему-то учить, то учите сразу хорошему, а именно независимым блокам. И используйте ресет или нормализер.

 +3

Ответить



- НЛО прилетело и опубликовало эту надпись здесь

 Metaller

18 ноя 2013 в 10:13

Хорошее разу учить использовать normalize.css или Reset CSS на худой конец и объяснить в двух словах, зачем это нужно.

 0

Ответить

 Seldon

18 ноя 2013 в 10:23

Все написано достаточно плохо, чтобы можно было смело не рекомендовать данный материал к прочтению а тем более использованию как учебное пособие.

Простите. мне кажется человек который берется учить людей, должен сам являться экспертом в выбранной области и суметь обосновать свой подход.

Я же думаю что вы не сможете обосновать следующие строки

1) #wrapper { 2) #sitemap div { 3) form[name="search"] { 4) header {

Вот пожалуй 4 грубейшие ошибки вашего кода. Вы учите людей делать не правильно, мне кажется нужно для начала самому немного подучиться

 0

Ответить

 RekrutUA

18 ноя 2013 в 18:04

можете посоветовать где смотреть как делать правильно?)

 +3

Ответить



- НЛО прилетело и опубликовало эту надпись здесь

 RekrutUA

18 ноя 2013 в 19:36

по ссылке выдает одни warning-и насчет ID и опасности изменения размеров, ничего толкового.

Рекомендации гугла читали, давно.

Человек выше написал, что там «все достаточно плохо», если имелось ввиду, что «все» это про кучу ID (и то что пара блоков не выделена в отдельные классы), то ок, проехали.

 +1

Ответить

 Seldon

19 ноя 2013 в 13:02

Нигде, нету учебника который научит вас делать хорошо. Нужно постоянно читать W3C документацию, читать тематические форумы, твиттеры, посты, блоги, иногда не плохо посмотреть какое-нибудь видео с конференций. Только самостоятельный, постоянный поиск информации.

0 Ответить



0 Аноним 18 ноя 2013 в 10:29

Если вы считаете, что это пример хорошей верстки – не продолжайте писать статью.

0 Ответить



0 gakot 18 ноя 2013 в 10:42

В начальной картинке разметки небольшой обман, футер не входит во врапер. А вообще для начала хорошо, для тех кто ни когда не верстал весьма полезно, только теперь прочтите комментарии и напишите статью по потенциальным проблемам которые вызывает ваша верстка, будет хорошая статья.

+1 Ответить



0 RekrutUA 18 ноя 2013 в 18:44

а зачем он должен входить? Что от этого изменится?

Дабы не быть пустословом, пара примеров:

rkz.ru/

www.pushe.ru/

0 Ответить



0 RekrutUA 18 ноя 2013 в 18:49

Правый клик «просмотр кода страницы» и скажите, что Хабр верстали нубы.)
Причем на хабре даже как-то наоборот все сделали – вложили layout во wrapper, обычно наоборот всегда замечал.

0 Ответить



0 gakot 18 ноя 2013 в 19:20

Должен или не должен это дело вкуса, другое дело что на схеме входит, а в верстке не входит.

+1 Ответить



○  **dikkini**
18 ноя 2013 в 11:07

Жду следующую статью, так как люблю bootstrap!

◆ +1 Ответить



○  **sunnybear**
18 ноя 2013 в 11:12

Использование селекторов вида figure div крайне негативно отражается на производительности. Браузеры разбирают селекторы справа налево, а div на странице может быть тысячи

◆ -1 Ответить



○  **wwwsevolod**
18 ноя 2013 в 11:15

ruu.sh/51TLJ.png
не учите людей плохому.

◆ +1 Ответить



○  **u0gurT**
18 ноя 2013 в 11:25

Конец 2013 года, а вы верстаете сайт и ограничиваете его статичными размерами, указывая размеры и отступы в пикселях.

Вы серьезно? Мне кажется, таких верстальщиков нужно перепрофилировать, и либо из них будет хороший «фронт-энд-разработчик», либо «вон из профессии».

Крайне часто приходится сталкиваться с верстальщиками, и давно я такого кода не видел.

◆ -3 Ответить



○ НЛО прилетело и опубликовало эту надпись здесь

○  **VolCh**
18 ноя 2013 в 23:54

Не учитывает различные размеры и разрешения экранов.

◆ 0 Ответить



○ НЛО прилетело и опубликовало эту надпись здесь

- НЛО прилетело и опубликовало эту надпись здесь

•  VolCh
19 ноя 2013 в 00:26

Градусы :) Reference pixel в терминологии CSS. Но в любом случае по-моему лучше использовать сантиметры или дюймы – вероятность того что браузер интерпритирует их с учетом dpi и обычного расстояния выше чем у пикселя. Субъективно. В худшем случае браузер будет считать дюйм равным 96 пикселям, то есть разницы с указанием в пикселях не будет, но может посчитать и с учетом конкретного евайса, в то время как пиксели отображать 1:1.

 -1 Ответить



- НЛО прилетело и опубликовало эту надпись здесь

•  VolCh
19 ноя 2013 в 03:47

Кто сказал, что 10cm это 10 физических сантиметров? Это где-то 8,2 градуса. На мобильном устройстве линейный размер будет один, на экране проектора совсем другой, но угловой должен быть одинаковый.

 0 Ответить



- НЛО прилетело и опубликовало эту надпись здесь

•  VolCh
19 ноя 2013 в 03:43

Потому что по стандарту, на который вы ссылаетесь не дюйм равен 96 (физическими) пикселям, а (логический) пиксель равен 1/96 дюйма.

 0 Ответить



- НЛО прилетело и опубликовало эту надпись здесь

•  VolCh
19 ноя 2013 в 04:12

:)

In particular, in previous versions of CSS the pixel unit and the physical units were not related by a fixed ratio: the physical units were always tied to their physical measurements while the pixel unit would vary to most closely match the reference pixel. (This change was made because too much existing content relies on the assumption of 96dpi, and breaking that assumption breaks the content.)

Опять из-за ВС всё зарубили :(

0 Ответить



- НЛО прилетело и опубликовало эту надпись здесь

VolCh
19 ноя 2013 в 10:35

Но вместо того, чтобы исправлять браузеры, стали исправлять стандарт.

0 Ответить



- НЛО прилетело и опубликовало эту надпись здесь

VolCh
19 ноя 2013 в 11:17

The following Win32 API functions are useful for retrieving information about the current display setting:
GetDeviceCaps Retrieves device-specific information for the specified device.
GetSystemMetrics Retrieves the specified metric or system configuration setting.
SystemParametersInfo Retrieves or sets the value of one of the system-wide parameters.

0 Ответить



- НЛО прилетело и опубликовало эту надпись здесь

VolCh
19 ноя 2013 в 14:10

Тем не менее возможности есть, но ею практически никто не пользуется.

0 Ответить



- НЛО прилетело и опубликовало эту надпись здесь

u0gurT
25 ноя 2013 в 11:31

Проценты.

0 Ответить



garmoncheg
18 ноя 2013 в 11:34

Огромное СПАСИБО! Как раз хочу научить жену такому типу работы... Ждем продолжения, и подобных статей.

+1 Ответить

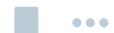


...

pvasili
18 ноя 2013 в 11:35

Т.с. поделитесь, пожалуйста, русской гарнитурой oswald.

0 Ответить



...

ded_Sergei
18 ноя 2013 в 11:44

Я всегда считал, что критиковать всегда проще, чем что то делать.

Есть замечания по верстке... но я сам отнюдь не специалист... подожду развернутых комментариев от профессионалов.

Автору спасибо.

0 Ответить



...

ilyautkin
18 ноя 2013 в 11:53

Кстати, я всегда считал, что DOCTYPE принято писать касом, а в последнее время чаще вижу написание «doctype». Это такой тренд?

+1 Ответить



...

Vend3tta
18 ноя 2013 в 16:42

Браузеру совершенно наплевать на то, какими буквами вы напишете доктайп – заглавными или строчными, можете хоть <!DocTyPe hTmL> написать – никто не обидится.

Главное, чтобы доктайп в принципе был указан, и перед ним в коде не было никаких непробельных символов.

+1 Ответить



...

ilyautkin
18 ноя 2013 в 17:30

У меня даже Geany по-разному подсвечивает:

<pre> 1 <!DOCTYPE html> 2 <html> 3 <head> 4 <meta chara 5 </head> 6 <body> </pre>	<pre> 1 <!doctype html> 2 <html> 3 <head> 4 <meta charset="utf-8"> 5 </head> 6 <body> </pre>
--	--

 +1

Ответить



...



Volch

19 ноя 2013 в 00:05



In other words, <!DOCTYPE html>, case-insensitively.

www.w3.org/TR/html5/syntax.html#the-doctype

 0

Ответить



...



derSmoll

18 ноя 2013 в 12:25

В качестве пустых картинок из макета будем использовать однопиксельное серое изображение, которое будем растягивать по необходимости

Серьезно?

 +3

Ответить



...



Win32Sector

18 ноя 2013 в 13:06

А как нужно? Я просто не в курсе. Только начинаю изучать верстку.

 0

Ответить



...



Mirantus

18 ноя 2013 в 13:25



Масштабирование изображений в html через стили или атрибуты считается плохим тоном. Правильно загружать изображение в масштабе 1 к 1 и прописывать правильные размеры через атрибуты width и height для тега img. Данная техника была применена тут только в качестве «заглушки», т.к. в макете не нарисованы настоящие картинки.

 0

Ответить



...



Win32Sector

18 ноя 2013 в 13:37



Ну то есть, в качестве заглушки лучше использовать нечто вроде pic.png и css вроде

```
#pic {  
    height:96px;  
    width:96px;  
    background-image: url ('..images/pic.png');
```

```
background-repeat: no-repeat;  
}
```

?

0 Ответить



...

○ Mirantus
18 ноя 2013 в 13:42

Нет, в данном случае, например вместо заглушки для карты

нужно было вставить ссылку на отдельную картинку соответствующего размера

Пусть даже эта картинка была бы просто серая.
Использование CSS здесь не нужно.
Я обязательно постараюсь избегать заглушек в следующих статьях.

0 +1 Ответить



...

○ kovalevsky
18 ноя 2013 в 13:41

в таких случаях лучше пользоваться placeholder.it/, например.
это лучше в качестве заглушки

0 +3 Ответить



...

○ navstyachka
18 ноя 2013 в 16:33

Согласна полностью – быстрее, удобнее. Кроме того, есть отличный сервис logempixel.com – там есть разделы по тематикам.

И для любителей «ня» – placekitten.com :)

0 +2 Ответить



...

○ НЛО прилетело и опубликовало эту надпись здесь

○ НЛО прилетело и опубликовало эту надпись здесь

○ hasalex
18 ноя 2013 в 12:50

Спасибо за статью.

Тема и формат интересные! во-первых, вполне разжевано, во-вторых – понятно для начинающих, в-третьих – есть замечания по сути. Будем ждать ответов на замечания – раз, продолжения статей – два, новых хороших комментариев – три!

0 Ответить



...

sevka_fedoroff
18 ноя 2013 в 13:43

Мне статья понравилась. У меня знания по верстке остались на уровне 2000 года. А теперь я понял, как вообще люди это делают последние лет 10. Ну и узнал про новые теги HTML5.

+2 Ответить



...

coldman
18 ноя 2013 в 14:18

Небольшое замечание, «LOREM IPSUM» написание текстов в верхнем регистре, имеет смысл подкреплять CSS правилом `text-transform:uppercase;`. Не нужно будет помнить, что по дизайну должен быть «`uppercase`» в конкретных местах и следить за этим. Рано или поздно, программист, или контент менеджер, кастомер наберет текст не «`uppercase`». и дизайнеры с тестерами будут сеять панику))

+2 Ответить



...

agentx001
18 ноя 2013 в 14:28

А мне статья понравилась, хотя я и полный профан в области верстки:) Так как много критики – не могли бы эти самые «критиканты» подкинуть годных статей о том, как правильно верстать?

0 Ответить



...

Priilepsky
18 ноя 2013 в 14:30

Статья не плохая, но есть несколько неясностей.

Пожалуйста описывайте для чего и почему используете, например, `figure`.

Почему карточки сотрудников это `figure`, а не `article`, или просто `div`? Я не понимаю.

+1 Ответить



...

nazagrc
18 ноя 2013 в 15:21

Не люблю я избыточные обертки, списки внутри навигации, когда достаточно просто ссылок внутри `nav`, `#footer` внутри `footer` (почему не `body > header`? Никакого

излишества + сразу понятно, где элемент и для чего он), и прочих олдскульных техник. Сейчас браузеры вполне позволяют делать макеты с несколькими колонками без лишних оберток.

0 Ответить



RekrutUA
18 ноя 2013 в 18:33

Скажешь это заказчику, когда он попросит кроссбраузерность от IE7. Причем выполнить верстку будет значительно быстрее как просят, нежели объяснять заказчику, что даже мамонты давно перешли на IE8+.

0 Ответить



nazagrc
18 ноя 2013 в 18:43

Не везет вам с заказчиками :(

Мои даже об IE 9 вспоминают очень редко. Не вижу смысла потакать выпендрежам старой версии браузера, когда все браузеры ведут себя более или менее по стандартам, а он, видите ли, имеет свои стандарты.

Мое мнение – HTML должен быть для контента, как он и задуман. То есть в идеале в HTML не должно быть ничего того, что не является контентом – излишних оберток, `<div class="clearfix"></div>`, и прочего подобного. И в абсолютном большинстве мне удается этого добиться с помощью CSS, для визуальных фишечек обычно достаточно псевдо-элементов (стрелочки, фон шапки на всю ширину страницы при фиксированной ширине содержимого, и т.д.).

Я понимаю, что у вас свои особенности – но в вебе такая консервативность вредна, и если учить людей верстке – то современной.

0 Ответить



RekrutUA
18 ноя 2013 в 19:01

" и если учить людей верстке – то современной."

В каком смысле?

Предлагаете забить на армию пользователей с IE8-9 и сразу хреначить разбивку блоков с помощью flexible layout и прочими прелестями?) Мне пока даже во сне такое не снилось.

Есть заказчик, есть его бабки и есть конкретные требования, личные предпочтения в любви к особым инструментам тут не имеют смысла. ИМХО. (исключением являются случаи когда заказчик ничерта не понимает в браузерах и ТЗ составил по образцу которое нашел в сети).

0 +2 Ответить



nazagrc
18 ноя 2013 в 19:09

| Предлагаете забить на армию пользователей с IE8-9 и сразу хреначить разбивку блоков с помощью `flexible layout` и прочими прелестями?)

Да! Да! Да!

Я такое уже практикую на рабочих сайтах понемногу (о `display: flex`), а псевдоэлементы уже давно стали частью обычной верстки.

Многие говорят об MVC, и в это же время используют HTML для формирования интерфейса. HTML – просто семантические данные, как они отображаются должен решать CSS. Если это не осуществимо на 100% сейчас – это не значит, что не стоит к этому стремиться.

| Есть заказчик, есть его бабки и есть конкретные требования

Если вы фрилансер – я вас не понимаю, совсем, а если вы работаете в компании – то я бы на вашем месте искал другое место, где народ понимает скорость развития веба и стоимость поддержки монстров каменного века.

0 Ответить



o VoLCh
19 ноября 2013 в 00:11

| где народ понимает… стоимость поддержки монстров каменного века.

А если народ также понимает, что эта стоимость многократно окупается?

0 Ответить



o пазагрс
19 ноября 2013 в 00:22

Такое может быть, но:

- а) далеко не всегда
- б) со временем поддержка становится всё сложнее, а выгоды всё меньше
- в) вы заставляете пользователей грузить лишний трафик и выполнять код с обходами и трюками для старых версий, что сложно назвать выгодой
- г) вы сдерживаите развитие интернета (если бы, как Google, все поддерживали только последние две версии, читали бы ли мы с вами об поддержке IE + в этом треде?)

Обновление до новой версии доступно всегда, и оно требует минимальных усилий со стороны пользователя.

Понятно, что есть разные крайности, и окупаемость старых версий имеет место быть, но это слишком далеко от майнстрима.

 0 Ответить VolCh
19 ноя 2013 в 00:28

Обновление до новой версии доступно всегда, и оно требует минимальных усилий со стороны пользователя.

Увы, нет. Некоторые пользователи и слова-то такого «обновление» применительно к компьютерам, а то и телефонам не знают

 0 Ответить dordzhiev
18 ноя 2013 в 21:30

Не вижу смысла потакать выпендрежам старой версии браузера, когда все браузеры ведут себя более или менее по стандартам, а он, видите ли, имеет свои стандарты.

Во времена иеб стандартов то и не было

 -1 Ответить nazagrc
18 ноя 2013 в 22:55

Но сейчас другие времена, есть большой выбор браузеров – простых, сложных, функциональных, надежных, открытых, красивых – выбирайте любой. Все из них стараются соответствовать стандартам, которые принимаются общими усилиями, и вдруг появляется новая версия «браузера», у которого пачка своих новых и интересных «стандартов».

Даже IE 11 не понимает стандартизированный `display: flex;`, у него свой гибридный `display : -ms-flexbox;`, и судя по всему так будет всегда.

 -1 Ответить dordzhiev
18 ноя 2013 в 23:20

CSS flexbox layout пока находится на стадии рекомендации, на W3C есть черновик. У chrome до этого лета тоже был префикс (`display: -webkit-flex`). Так что не придумывайте про стандарты.

 +1 Ответить nazagrc
18 ноя 2013 в 23:28

Последняя версия рекомендации от 18 сентября прошлого года, то есть более чем за год до релиза Windows 8.1, а в браузере не реализовано. Всё это создает ненужную энтропию. Если за год выйдет десяток версий Chrome и пяток Firefox/Орга, то IE, который устарел ещё до релиза мы будем (или нет:)) поддерживать ещё долго, ибо они не включат поддержку правильного синтаксиса обновлением. В этом вся проблема – скорость и кривизна его развития никак не вписывается в современный веб.

◆ 0 Ответить



◦ **dordzhiev**
19 ноя 2013 в 08:07

18 сентября написана рекомендация. Сам черновик датирован 30 октября этого года.
Но было бы неплохо выпускать новые версии IE почаще. Раза в 2 эдак.

◆ 0 Ответить



◦ НЛО прилетело и опубликовало эту надпись здесь

◦ **nazagrc**
18 ноя 2013 в 18:45

Приводите в пример Google – стабильная + предыдущая версия браузера. Всё, что раньше работать может, но совсем не обязательно. Не хотите обновляться – сами виноваты.

◆ -1 Ответить



◦ **RekrutUA**
18 ноя 2013 в 19:08

В данном случае это очень долгая и холиварная тема. Иногда даже кажется, что если бы браузеры стоили пусты и 1\$, а каждое обновление .99c, то народ бы охотнее обновлялся. Что-то вроде споров про Мак и ПК
demotivation.me/0g5r9lo7c73fpic.html

◆ +1 Ответить



◦ **KonstRuctor**
18 ноя 2013 в 15:23

В рамках данной статьи было бы очень интересно почитать про такую тему, с аватарками сотрудников.
А именно – у сотрудников могут быть длинные имена-фамилии, которые будут занимать несколько строк и которые (Череззаборонагозадерищенский) точно не влезут по ширине.
То же самое с должностью (главный помощник старшего менеджера по закупкам нового перспективного оборудования).

0 Ответить

 **pazagrs**
18 ноя 2013 в 15:36

Как вариант – просто обрезать:
`text-overflow: ellipsis;`

1 Ответить

 **jab**
18 ноя 2013 в 20:48

В английском языке имена/фамилии и должности короче.

-1 Ответить

 **olegkrasnov**
24 ноя 2013 в 14:18

`­`

0 Ответить

 **SunDrop**
18 ноя 2013 в 15:29

Спасибо!

0 Ответить

 **VYakushev**
18 ноя 2013 в 15:39

Спасибо! Как раз актуальный вопрос для меня.

0 Ответить

 **radist2s**
18 ноя 2013 в 17:33

Да ладно вам, вы на автора накинулись за селекторы айдишниками и тегами, а между тем цель статьи в том, чтобы показать новичкам что вообще из себя представляет реальная верстка. Понятное дело, что потом авторе скажет, мол, «помните как мы делали в первой статье – так вот, так делать не нужно, потому-то потому-то». А в целом, вполне себе годная статья, вот только сразу все охватить сложно, и по-хорошему такие статьи сначала надо показывать коллегам, а затем публиковать.

Я бы правда добавил в статью еще пару строк про составные селекторы, а то в свое время для меня было просто открытием, что можно делать так:

```
.link.bold + img.thumbnail{ }
```

+2 Ответить



stranger777
7 окт 2015 в 14:26

В CSS3 можно даже так:

```
/*
The following selector represents an HTML anchor a with an href attribute whose v
Обработает все теги a с атрибутом href при условии, что значение href заканчивает
*/
a[href$=".html"]
```

W3C Recommendation 29 September 2011

Спецификации и документация браузеров (и не только браузеров, а также языков и инструментов) – наше всё. Был бы ещё такой хитрый метод насильного обновления updateForce(); в JS со своей реализацией под каждый браузер – всем бы легче жилось. Но нет, занимаются WebGL и прочими красивостями.

0 Ответить



RekrutUA
18 ноя 2013 в 18:26

HOME

ABOUT US

Кэпс в html конкретно выдает уровень писанины, ну и с ID перебор какой-то.

Сам верстальщик, правда с мелким опытом.

Забавные комментарии у критиков, которые даже толком не «тыкнули» в ошибки.

Люди, эти статьи читает очень много новичков, если не хотите чтобы завтра в вашей фирме кто-то так верстал, то будьте добры, давайте ссылки на материалы/книги которые вы считаете эталоном по фронт-энду.

+3 Ответить



p4p
18 ноя 2013 в 19:22

А ну теперь понятно...

0 Ответить



ISpy

 18 ноя 2013 в 21:15 

Не критика, а просто комментарии и мои предпочтения в верстке.

- DOCTYPE пишу в верхнем регистре. Для html5 не принципиально, но xml-парсер, например, выдаст ошибку.
- Если используете html5, то пишите meta , link короче. Также, лучше закрывать одиночные теги.

```
<link rel="stylesheet" href="default.css" />
<meta charset="utf-8" />
```

- ИЕ6 конечно умер, но я стараюсь минимально его поддерживать, поэтому вместо селекторов атрибута типа [type="text"] и других подобных вещей лучше добавлю лишний класс. Вообще, часто приходится отказываться от удобных решений в угоду совместимости.
- Логично было бы поместить nav и #heading в header . Также footer по схеме на картинке, должен быть внутри #wrapper :

```
<body>
  <div id="wrapper">
    <header>
      <nav></nav>
      <div id="heading"></div>
    </header>
    <aside></aside>
    <section></section>
    <footer></footer>
  </div>
</body>
```

- Нужно установить :visited -цвет ссылок меню, иначе их цвет может измениться, после перехода пользователя.

```
.top-menu a,
.top-menu a:visited {
  color: #b2b2b2;
}
```

- Используя свойство background , обратите внимание, что если вы устанавливаете только, например цвет (background: red;), то сбрасываются все другие фоновые установки у данного элемента (image , position и т.п.). Поэтому, если нужно установить только одно свойство, пишите лучше не background: red; , а конкретное

свойство:

```
background-color: red;
```

– То же, кстати, касается `font` – используя его, но не перечисляя всё, вы рискуете, что часть свойств вернется к значениям по-умолчанию, независимо от того, что ранее вы установили их для этого элемента. Здесь можно увидеть, что я имею ввиду. У `background` то же поведение, поэтому лучше взять за правило использовать по-необходимости частные свойства, а не только общие.

– Нигде не увидел борьбы с `floating`-багом. Нужно либо добавлять `clear`-элемент после группы плавающих элементов, либо присваивать их контейнеру `overflow: hidden;`, либо использовать какой-то еще способ. Я бы еще и для старого IE пофиксил с помощью установки `hasLayout` (например `zoom: 1;`).

Ну и общие замечания – по излишнему использованию `id`, отсутствию независимости блоков и соответственно излишнее увлечение селекторами тегов. Не бойтесь использовать классы в любом количестве. Кстати тоже посоветую прочитать про БЭМ. Можно извлечь полезные мысли, хотя полностью концепция слишком громоздкая имхо.

P.S. Комментария моего комментария приветствуются :)

 +7 Ответить

 ...

○ НЛО прилетело и опубликовало эту надпись здесь

 **ISpy**
19 ноя 2013 в 04:20

▲

Boy, уровень занудства over 9000, я даже запутался :)

В HTML(5) действительно нельзя реально «закрыть» одиночный (`void`) тег. Закрывающей слеш поставить можно, но это ни на что не влияет и не делает одиночный тег самозакрытым. Но совет был дан исходя из логики XML (чтобы получить валидный документ), где постановка закрывающего слеша в открывающем теге закрывает его, а одиночных тегов нет вообще. Поэтому и написал «закрывать (логика XML) одиночные (логика HTML) теги».

P.S. Написал текст выше и понял, что это выглядит как бред (:

 0 Ответить

 ...

○ НЛО прилетело и опубликовало эту надпись здесь

- НЛО прилетело и опубликовало эту надпись здесь

•  **ISpy**
19 ноя 2013 в 04:22

Собственно что и имелось ввиду.

0 Ответить



•  **exeto**
18 ноя 2013 в 22:34

Везде где подразумевается использовать только заглавные буквы, лучше это делать средствами CSS

+2 Ответить



•  **псих**
19 ноя 2013 в 00:14

Я довольно далек от верстки, и возможно сейчас спрошу глупость, но все же. Вот есть у вас элементы, которые будут на каждой странице (шапка, меню, логотип и т.д.). Вы их вставили прямо в страницу. Я так понимаю в остальные страницы все это будет копипаститься? И если что-то захочется поменять среди этих элементов, придется обойти все страницы? Нельзя ли вынести все это в одно место, откуда будет подтягиваться для других страниц? Или я чего-то не понимаю?

0 Ответить



•  **VolCh**
19 ноя 2013 в 00:30

В рамках верстки подобные проблемы очень редко решаются. Для этого обычно используют серверный софт, на который такую верстку «натягивают».

+2 Ответить



•  **Mirantus**
19 ноя 2013 в 06:07

Попробуйте почитать про шаблонизаторы. Я думаю, что вы откроете для себя новый удивительный мир :)

-1 Ответить



• НЛО прилетело и опубликовало эту надпись здесь

•  **Mirantus**
24 ноя 2013 в 18:00

Вашему комментарию не хватает только объяснения почему центрирование блока через left, right лучше, чем через margin: auto.

0

[Ответить](#)

•••

Только полноправные пользователи могут оставлять комментарии. Войдите, пожалуйста.

Публикации

[ЛУЧШИЕ ЗА СУТКИ](#) [ПОХОЖИЕ](#)

DrMefist0

23 часа назад

Исследуем саундбар Yamaha YAS-109

Сложный

9 мин

8.5K

[Туториал](#)

+132

61

28



SergeyAstanin

21 час назад

Анимация на колесных дисках с беспроводной передачей питания

Средний

6 мин

5.8K

[Туториал](#)

+62

36

28



0_kot_diteected

вчера в 20:19

Пилотируемый полет США на Марс в 2039

Средний

14 мин

5.6K

[Из песочницы](#)

+46

28

17



PatientZero

19 часов назад

Как у меня украли авторство патча

Средний 5 мин 12K

Ретроспектива

Перевод

+40

18

81

 CyberexTech
4 часа назад

Как я делал солнечную зарядку для своего моноколеса и вот что получилось

Простой 6 мин 1.7K

Из песочницы

+28

14

16

 deema35
11 часов назад

Сборка прошивки из исходников для Orange PI i96(Orange PI 2g-iot)

Простой 16 мин 2.1K

Туториал

+27

37

10

 ru_vds
вчера в 20:00

Минимум менеджеров, максимум инженеров: как Threads* удалось выпустить за пять месяцев

Простой 13 мин 2.6K

Обзор

Перевод

+27

20

2

 Ikgamag
6 часов назад

React 18: что поменялось

Простой 19 мин 1.4K

Кейс

+26

18

1



shiru8bit

4 часа назад

Самодельные портативные игровые консоли и ESPboy

 Простой
 22 мин
 1.2K

Обзор

+23

20

4



eapotarov

4 часа назад

Как не превратить облако в денежную дыру

 5 мин
 498

Туториал

+17

9

0

Как устроен электронный документооборот в ЕВРАЗе

Турбо

Показать еще

МИНУТОЧКУ ВНИМАНИЯ



Интересно



Турбо

Глупым вопросам и ошибкам — быть! IT-менторство на ХК

Чем заняться в 1С опытному айтишнику?

ЗАКАЗЫ

Настроить сайт на wordpress

10 руб./за проект · 2 отклика · 22 просмотра

React frontend

1500 руб./в час · 9 откликов · 58 просмотров

Привязать приложение на базе Tuya к Яндекс.Алисе

20000 руб./за проект · 17 просмотров

Шаблонизация и программирование под 1с Битрикс

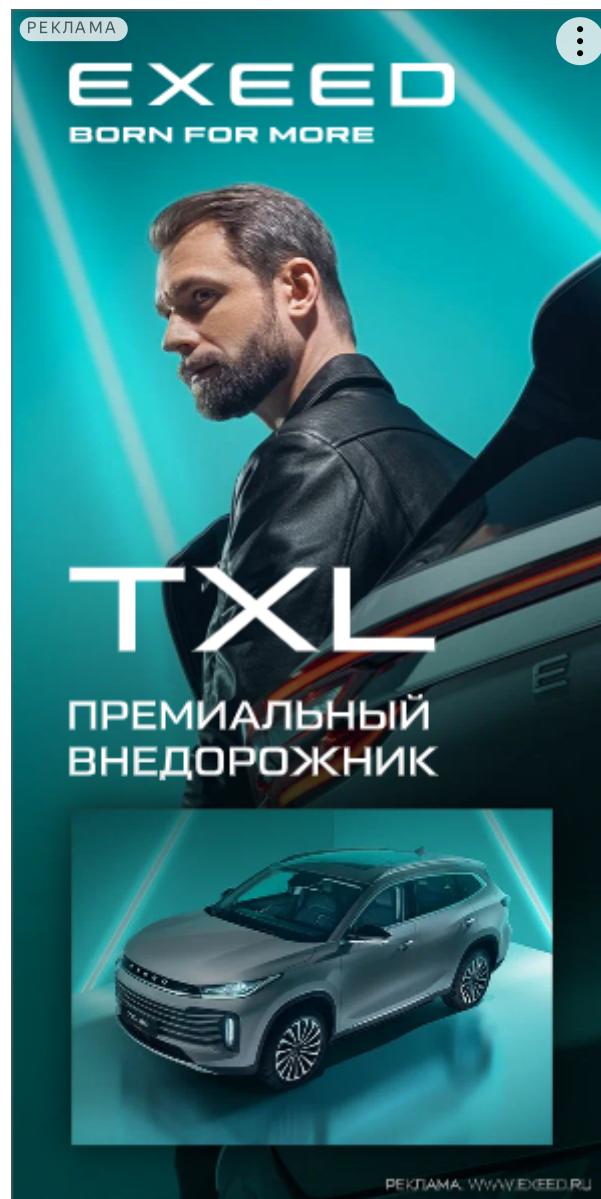
30000 руб./за проект · 6 откликов · 22 просмотра

Сверстать сайт

10000 руб./за проект · 70 откликов · 146 просмотров

Больше заказов на Хабр Фрилансе

Реклама



ЧИТАЮТ СЕЙЧАС

[Рег.ру объяснил обещание дать п***ы технической ошибкой](#)

2.2K 10

[Скам-империя братьев Дуровых](#)

35K 102

[Вышла Counter-Strike 2](#)

13K 28

[Valve проиграла в ЕС апелляцию по иску о геоблокировке в Steam и выплате штрафа в €1,6 млн](#)

1.6K 5

[Как управлять финансами, чтобы выйти на раннюю пенсию, а не в окно](#)

6.7K 51

[Как устроен электронный документооборот в ЕВРАЗе](#)

Турбо

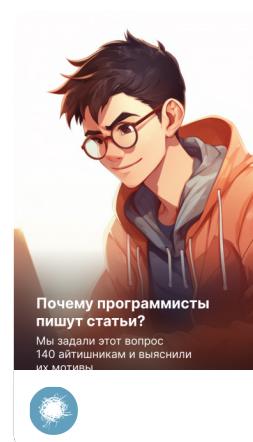
ИСТОРИИ



Лица SmartDev 2023



Топ-7 годноты из блогов компаний



Почему программисты пишут?



Знакомьтесь, слон



Учиться хорошо

Реклама



Ваш аккаунт

[Войти](#)[Регистрация](#)

Разделы

[Статьи](#)[Новости](#)[Хабы](#)[Компании](#)[Авторы](#)[Песочница](#)

Информация

[Устройство сайта](#)[Для авторов](#)[Для компаний](#)[Документы](#)[Соглашение](#)[Конфиденциальность](#)

Услуги

[Корпоративный блог](#)[Медийная реклама](#)[Нативные проекты](#)[Образовательные программы](#)[Стартапам](#)[Спецпроекты](#)[Настройка языка](#)[Техническая поддержка](#)[Вернуться на старую версию](#)