**Задание 1 Верстка простой страницы**

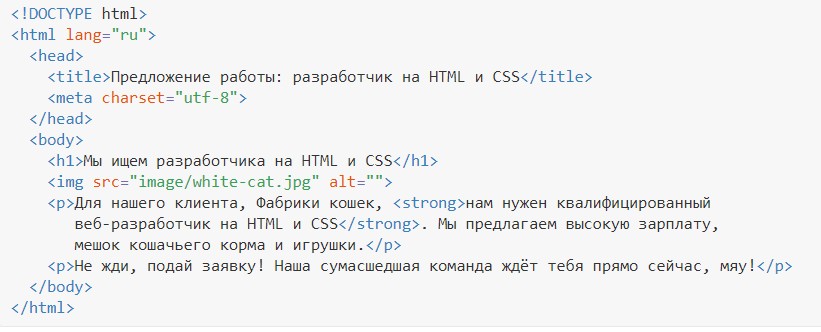
Нужно создать веб-страницу с размещением вакансии

Решение

Прежде чем создавать любые веб-страницы, нужно разделить содержимое на более мелкие компоненты по их важности.



Создаем каркас страницы:



Сохранить страницу.

**2. Создайте статью блога article.htm:**



Решение

Рассмотрим структуру статьи:

1. В статье, вероятно, наиболее важной информацией будет заголовок с текстом: «Джастин Бивер: С тех пор, как я узнал HTML, моя жизнь повернулась на 180 градусов».
2. Следующей важной информацией являются данные об авторе: «написал: Дамиан Вельгошик».
3. Далее идёт первый абзац текста, затем место для фото и описание.
4. Наконец, последний абзац содержит цитату: «Я, вероятно, спою об HTML в своём следующем альбоме».

При таком понимании иерархии наша информация в коде HTML должна быть организована следующим образом.

<!DOCTYPE html>

<html lang="ru">

<head>

<meta charset="utf-8">

<title>Джастин Бивер увлечён HTML</title>

</head>

<body>

<article>

</article>

</body>

</html>

Здесь использовали элемент <article>, который содержится в родителе <body>. Статьи состоят из заголовка и содержания, так что все теги представляющие заголовок и абзацы, естественно, будут дочерними <article>. Дополнительно к нашему списку элементов, который использовался до сих пор, добавится тег <header> — элемент, который маркирует заголовки.

Добавляем <header> как дочерний в <article>.

<!DOCTYPE html>

<html lang="ru">

<head>

<meta charset="utf-8">

<title>Джастин Бивер увлечён HTML</title>

</head>

<body>

<article>

<header>

</header>

</article>

</body>

</html>

Наш раздел заголовка готов, но что должно быть внутри <header>? Есть заголовок и автор. Будем использовать тег <h1> для заголовка текста, а <р> для автора. В HTML нет конкретного тега для автора, так что в этом случае используем <р> в качестве общего контейнера для текста.

В приведённом ниже коде добавлен теги <h1> и <p> в элемент <header>.

<!DOCTYPE html>

<html lang="ru">

<head>

<meta charset="utf-8">

<title>Джастин Бивер увлечён HTML</title>

</head>

<body>

<article>

<header>

<h1>Джастин Бивер: С тех пор, как я узнал HTML, моя жизнь повернулась на 180 градусов</h1>

<p>Написал: Дамиан Вельгошик</p>

</header>

</article>

</body>

</html>

Перейдём к вёрстке первого абзаца текста. Нужно избежать вложения абзаца в <header>. Это имеет смысл, потому что абзац должен быть частью <article>. Таким образом, добавляем первый <р> в <article> с тем же приоритетом что и <header>, но под ним.

<!DOCTYPE html>

<html lang="ru">

<head>

<meta charset="utf-8">

<title>Джастин Бивер увлечён HTML</title>

</head>

<body>

<article>

<header>

<h1>Джастин Бивер: С тех пор, как я узнал HTML, моя жизнь повернулась на 180 градусов</h1>

<p>Написал: Дамиан Вельгошик</p>

</header>

<p>Джастин Бивер сделал признание, которого никак не ожидали даже самые

большие поклонники талантливого музыканта и поэта. Молодой певец признался,

что когда он набрал свой первый тег, его жизнь стала легче. Из окружения

канадца сообщают, что частные наставники Бивера, Райан Лослинг и

Николас Крейт, часто гуляют по Лос-Анджелесу, споря об инструменте

HTML-валидатор.</p>

</article>

</body>

</html>

Далее добавляется фото плюс описание. Для этого типа контента, который связан со всем документом в целом, вроде фотографий, графиков или карт, использоватся тег <figure>. Следует отметить, что условия использования <figure>важно в том смысле, что вы можете использовать дополнительный элемент <figcaption>, который добавляет описание под изображением.

В результате получаем

<!DOCTYPE html>

<html lang="ru">

<head>

<meta charset="utf-8">

<title>Джастин Бивер увлечён HTML</title>

</head>

<body>

<article>

<header>

<h1>Джастин Бивер: С тех пор, как я узнал HTML, моя жизнь повернулась на 180 градусов</h1>

<p>Написал: Дамиан Вельгошик</p>

</header>

<p>Джастин Бивер сделал признание, которого никак не ожидали даже самые

большие поклонники талантливого музыканта и поэта. Молодой певец признался,

что когда он набрал свой первый тег, его жизнь стала легче. Из окружения

канадца сообщают, что частные наставники Бивера, Райан Лослинг и

Николас Крейт, часто гуляют по Лос-Анджелесу, споря об инструменте

HTML-валидатор.</p>

<figure>

<img src="cat.jpg" alt="Кот Джастина Бивера доволен">

<figcaption>Счастливый кот Джастина Бивера</figcaption>

</figure>

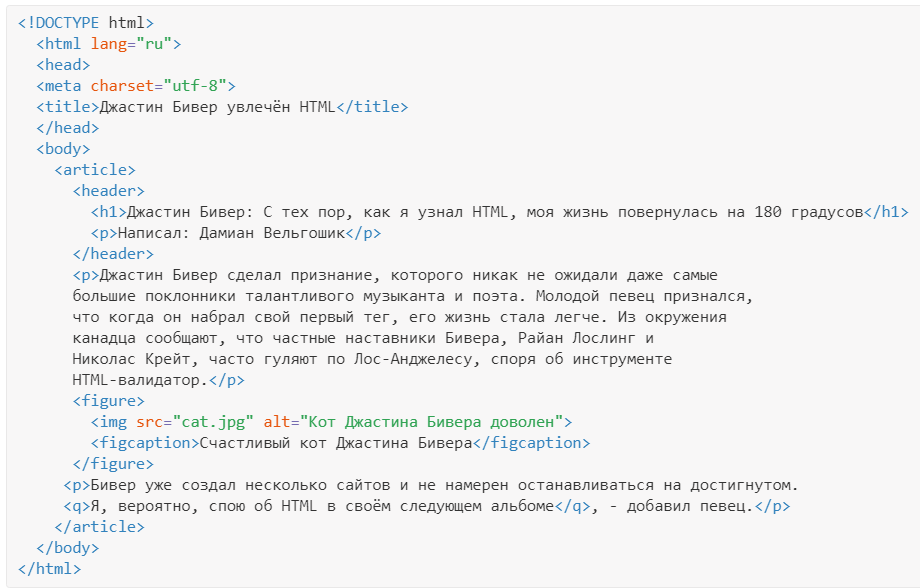
</article>

</body>

</html>

После добавленного изображения есть ещё один абзац, который надо вставить в статью. Обратите внимание, что этот абзац содержит цитату, «Я, вероятно, спою об HTML в своём следующем альбоме». Возможно в будущем кто-то станет искать цитаты Джастина Бивера и эта маркировка поможет им найти цитату быстрее. В противном случае поисковые системы всегда будут иметь дело с одним огромным куском текста.

Для указания что часть текста является цитатой используем тег <q> внутри нового абзаца.



Поменяем вид блога



Измените внешний вид с помощью таблицы стилей:

1. зелёный текст заголовка

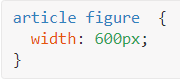


1. В разделе с информацией об авторе нужно, чтобы текст был белым на красном. Нужно изменить код так, чтобы селектор выше применялся только к <р> в <header>, который является «ребенком» <article>. Код должен отражать эту иерархию:

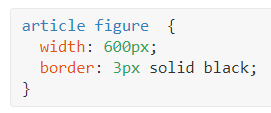


1. Перейдём к изображению в статье.

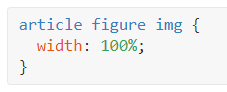
Размер этой статьи, скажем, должен быть 600 пикселей в ширину. Что для изображений есть соответствующий тег <figure>. Уточним наш код CSS, чтобы отразить это.



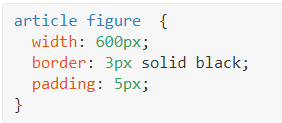
С помощью этого кода каждый тег <figure> в <article> будет иметь ширину 600 пикселей. Добавим границу рисунка



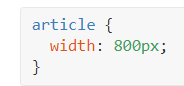
Граница отображается правильно, но изображение выводится за пределами наших 600 пикселей. Это потому, что установили ширину элемента <figure>, но тег <img> не имеет никакой фиксированной ширины и, таким образом, сохраняет свой исходный размер. Было бы неплохо, если бы изображение стало 100% от ширины родителя <figure>.



Зададим некоторый отступ или пространство между границей и изображением. Сделаем это, добавив свойство padding.



Текущий абзац распространяется почти на всю ширину окна браузера и поэтому не очень хорошо читаются, ограничим его до 800 пикселей.



Установим шрифт для <body>:



**Задание 3 Создание меню**

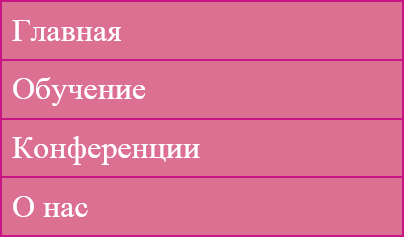
Популярной частью сайтов является меню. Реализуем меню. Начнём со следующего кода.



Меню будет состоять из четырёх элементов:

* Главная
* Обучение
* Конференции
* О нас

Вид меню



<nav> используется для определения всех видов навигационных функций на сайтах, которые содержат ссылки на внутреннюю или внешнюю информацию. Размещение <nav> в коде говорит «всё внутри <nav> будет использоваться для навигации по сайту». В <nav>  вставлен тег <ul> с несколькими тегами <li>.

Добавим стили для nav

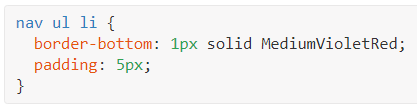


Здесь параметр list-style в значение none делает список без характерных маркеров.

Широкая область цвета большая. Нужно немного урезать её, используя тот же пример, что с рамкой вокруг изображения (padding).



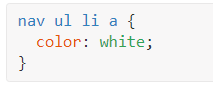
Каждому элементу в списке нужна своя граница.



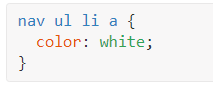
Но возникает проблема  в виде двойной линии в нижней части нашего меню. Это потому, что граница для меню добавилась к границе для последнего элемента, когда вставили bottom-border. Поэтому просто можно это убрать:



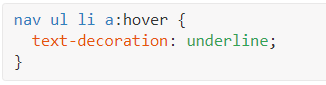
Изменим цвет шрифта на белый.



Уберем подчеркивание ссылок



Позволим ссылке выделиться при наведении на неё. Будем использовать псевдо-селектор под названием hover.

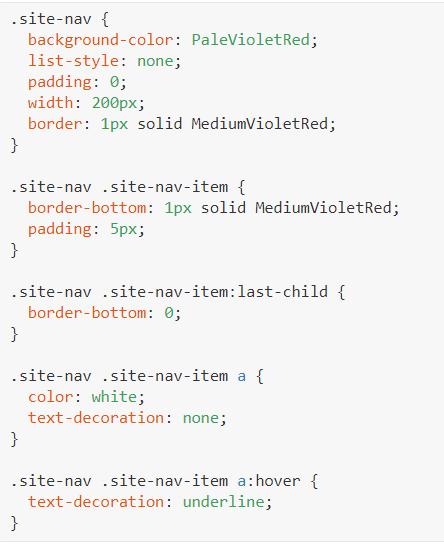


Измените html код страницы меню следующим образом, добавив классы



Класс всему контейнеру меню (тег <ul>), а каждому элементу в нём (тег <li>) «дочерний» класс.

Внесите изменения в стилевой файл



**Задание 4. Верстка формы**

Сформируйте каркас формы



Решение

браузер распознаёт три группы элементов:

* inline (строчные);
* block (блочные);
* inline-block (строчно-блочные).

Строчные элементы не вызывают перевода на новую строку и показывают один элемент рядом с другим по горизонтали. Блочные элементы устанавливаются как блоки, которые укладываются по верхней части друг друга и никогда не отображаются рядом по горизонтали.Строчно-блочные действуют в качестве строчных элементов (отображаются рядом друг с другом), но отличаются от них тем, что им можно задать размер. К примеру, поле <textarea> может отображаться как большой прямоугольник, но при этом занять пространство за пределами одной строки текста.

* Строчные — <span>, <em>, <strong>
* Блочные — <div>, <p>, <article>
* Строчно-блочные — <input>, <textarea>

По умолчанию, у строчных элементов стилевое свойство display установлено как inline. Для блочных элементов его значение block, а для строчно-блочных элементов inline-block. Однако можно изменить это поведение, добавив одну строку в CSS, например для span:

span {

display: block;

}

Введем в каркас формы тэг div:



Измените код для textarea

<textarea rows="10" cols="50" id="content" name="content"

placeholder="Введите здесь комментарий. Будьте внимательны и вежливы!"></textarea>

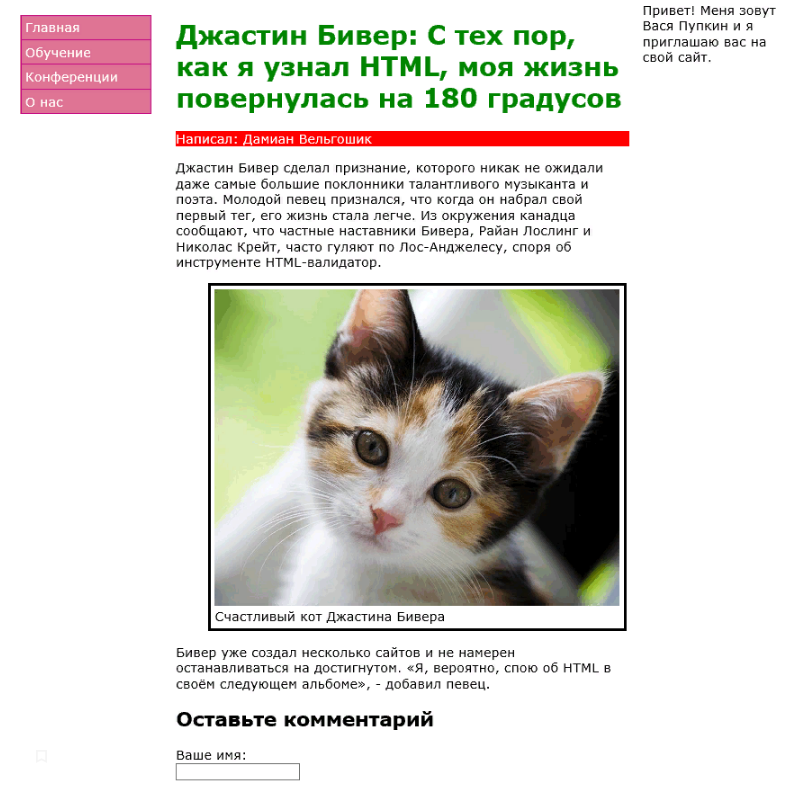
Создайте стилевой файл для формы:

Каждый элемент input содержащий атрибут type со значением text и email получит синюю рамку.



**Задание 4. Верстка многоколоночного макета**

Трёхколоночный макет, где меню слева, статья и форма комментариев по середине, а список материалов, связанных со статьёй, находится справа. Предположим, что вся страница занимает ширину 960 пикселей и расположена по центру.



Попробуем снова на рисунке определить функциональные части, которые отобразим в HTML. Есть левая колонка содержащая меню. В центральной колонке располагается основное содержание страницы. Правая колонка содержит краткий текст.

Начнём с контейнера, который содержит эти три колонки. Как уже отмечалось ранее, нужно чтобы все они занимали пространство не шире 960 пикселей. В HTML нет специальных тегов для такого условия, будем использовать тег <div>.

<div class="main-container"></div>

Даем <div> класс main-container, который говорит что это основной контейнер для остальных элементов сайта.

Теперь добавим место для меню, оно будет в <div class="site-menu">:

<div class="main-container">

<nav class="site-menu"></nav>

</div>

Двигаемся дальше и поместим контейнер для основного содержимого в среднюю колонку:

<div class="main-container">

<nav class="site-menu"></nav>

<div class="main-content"></div>

</div>

Правая колонка, боковая панель или «сайдбар»:

<div class="main-container">

<nav class="site-menu"></nav>

<div class="main-content"></div>

<aside class="sidebar"></aside>

</div>

Что должно содержаться в левой колонке? Как предполагает название, это должно быть меню из предыдущих упражнений:

<div class="main-container">

<nav class="site-menu">

вставьте сюда код меню

</nav>

<div class="main-content"></div>

<aside class="sidebar"></aside>

</div>

Средняя колонка будет содержать статью и форму комментариев:

<div class="main-container">

<nav class="site-menu">

вставьте сюда код меню

</nav>

<div class="main-content">

<article>вставьте сюда код статьи</article>

<form>вставьте сюда код формы</form>

</div>

<aside class="sidebar"></aside>

</div>

В правую колонку положим боковой элемент не связанный с основным содержанием, используя тег <aside>:

<div class="main-container">

<nav class="site-menu">

вставьте сюда код меню

</nav>

<div class="main-content">

<article>вставьте сюда код статьи</article>

<form>вставьте сюда код формы</form>

</div>

<aside class="sidebar">

<div>вставьте сюда код для сайдбара</div>

</aside>

</div>

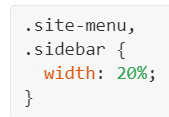
Изменяем стилевой файл: нужно установить максимальную ширину основного контейнера с классом main-container:



С помощью этого кода браузер берёт всё свободное пространство вокруг main-container и распределяет пространство поровну между двумя краями.

Пусть контейнер с меню занимает 20% от доступной ширины. Аналогичная ширина для правой колонки:

Поскольку левая и правая колонки содержат одинаковые свойства, мы можем сгруппировать их через запятую:



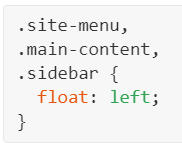
Теперь перейдём к средней колонке. Она займёт оставшуюся ширину (60%), так как две боковые колонки занимают в сумме 40%.



Контейнеры по-прежнему выводятся в виде блоков (один над другим). Чтобы установить их рядом друг с другом нужно дать им специальное свойство float. Используем его, чтобы сказать браузеру, что хотим сделать элемент X ближе к левому или правому краю контейнера, в котором он находится.



Можем сделать то же самое для колонок, которые хотим расположить «слева» по краю контейнера



Каждый элемент пытается «плыть» влево согласно приоритету в списке. В этом случае .site-menu занимает левый край, .main-content встаёт рядом в линию, чтобы заполнить крайнее левое пространство. Наконец, .sidebar заполняет последний «левый» край, завершая расстановку.

Добавим фоновый цвет к каждой колонке:

