



# Вёрстка – часть 1

# Ведущие вебинара

---



Ковальчук  
Дмитрий



Калин  
Антон



Захаров  
Зар



# Содержание

---

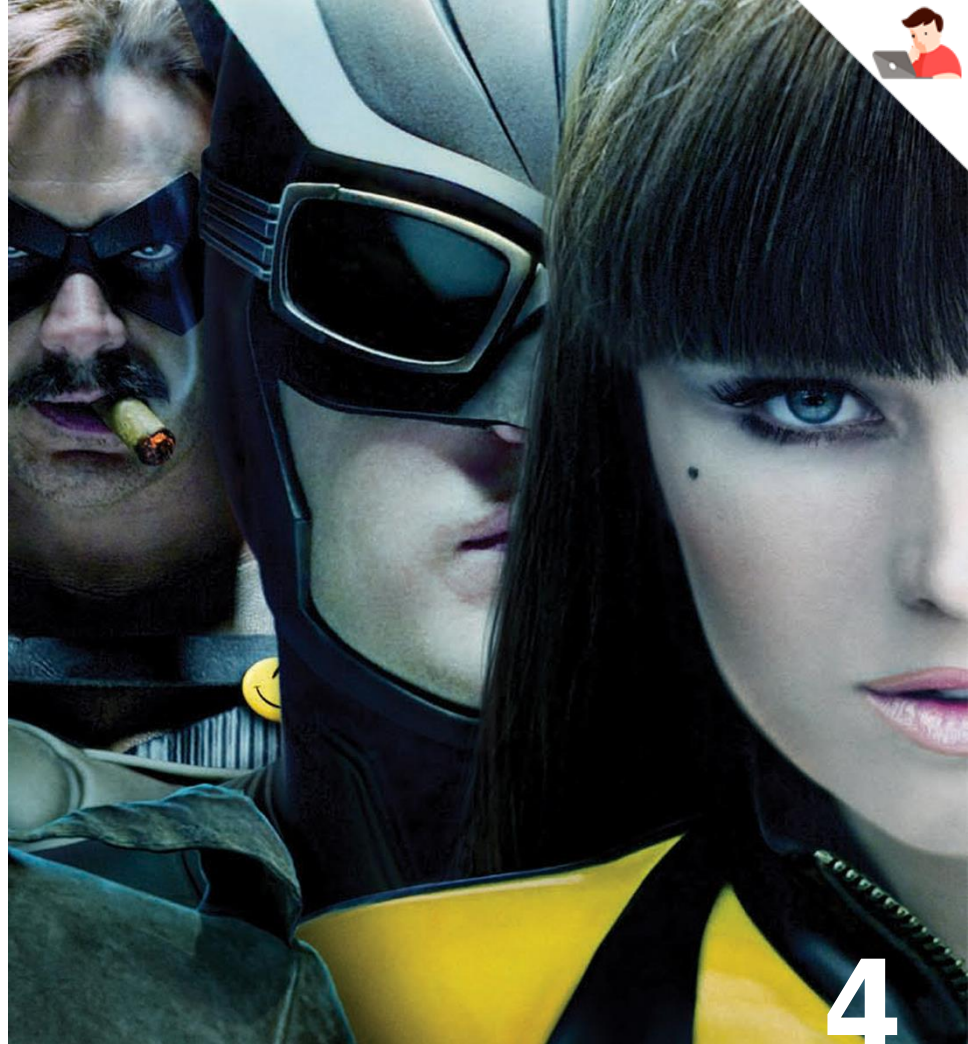
1. Viewport
2. Семантика в HTML
3. Войны значимости
4. Display
5. Сбросы CSS
6. Центрирование
7. Псевдоэлементы
8. Круглые картинки
9. Вопросы на засыпку

# Цель вебинара

---

Перед тем, как перейти к вет-методологии, flexbox и пр., нам необходимо убедиться, что мы с вами говорим на **одном языке**.

Начнём с базовых вещей, потом перейдём к более сложным.



A low-poly 3D rendered scene. On the left, a large, white, angular structure resembling a tent or a modern building stands on a flat, light-colored ground. To its right is a tall, grey, angular rock formation or monument. The background is a hazy, light grey sky. In the bottom right corner, there is a large red number '5'. There are also red L-shaped corner markers in the top right and bottom left corners.

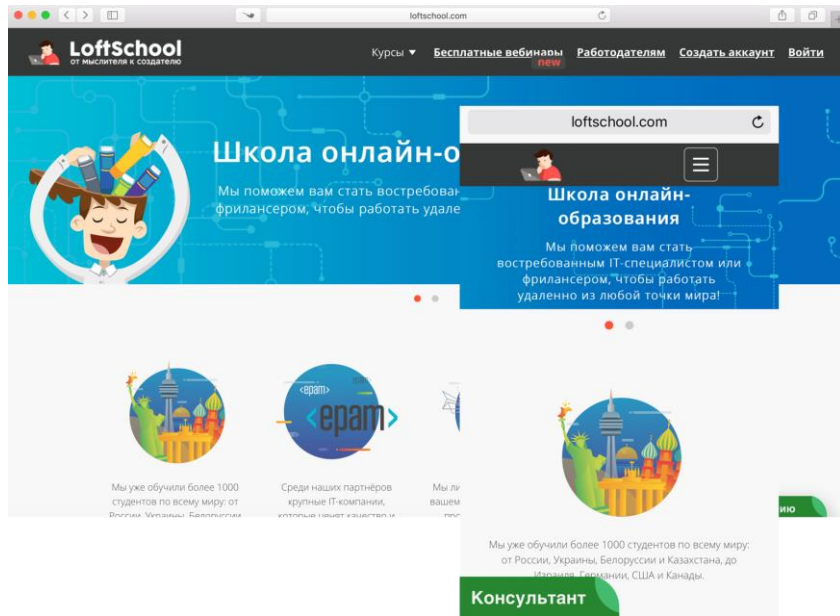
# Viewport

# Viewport для адаптивных сайтов



**Viewport** - это видимая пользователю область веб-страницы. Т.е. это то, что может увидеть пользователь, не прибегая к прокрутке.

```
<meta name="viewport"
content="width=device-width,
initial-scale=1">
```

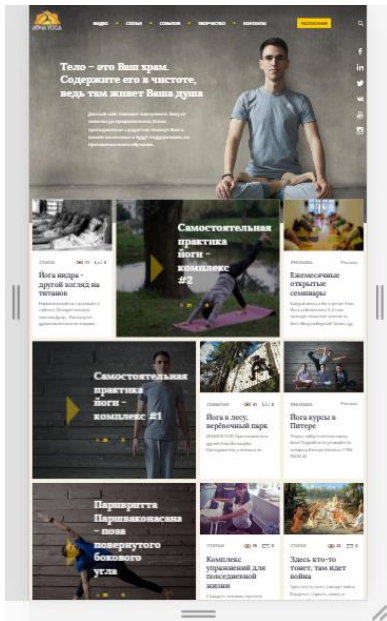




# Viewport для НЕ адаптивных сайтов

Если хотим, чтобы сайт масштабировался

```
<meta name="viewport"
content="width=device-width">
```



Если хотим отобразить в обычном масштабе, **не** уменьшенном

```
<meta name="viewport"
content="initial-scale=1">
```





# Семантика в HTML





# Общие рекомендации

---

## Спасибо, Кэп!

1. Каждый тег в вашей верстке используется только по своему прямому назначению.
2. HTML используется для создания структуры документа, а CSS для его оформления. В HTML документе никогда не должны встречаться встроенные стили. Забудьте про тег `<font>`.
3. Блочные теги не используются внутри строчных.
4. Ссылки не используются без текста внутри них.
5. Обязательные атрибуты, такие, как `alt` для картинок, всегда присутствуют.
6. Не злоупотребляйте тегами `<hr>` и `<br>`.
7. После дождя на улице мокро.

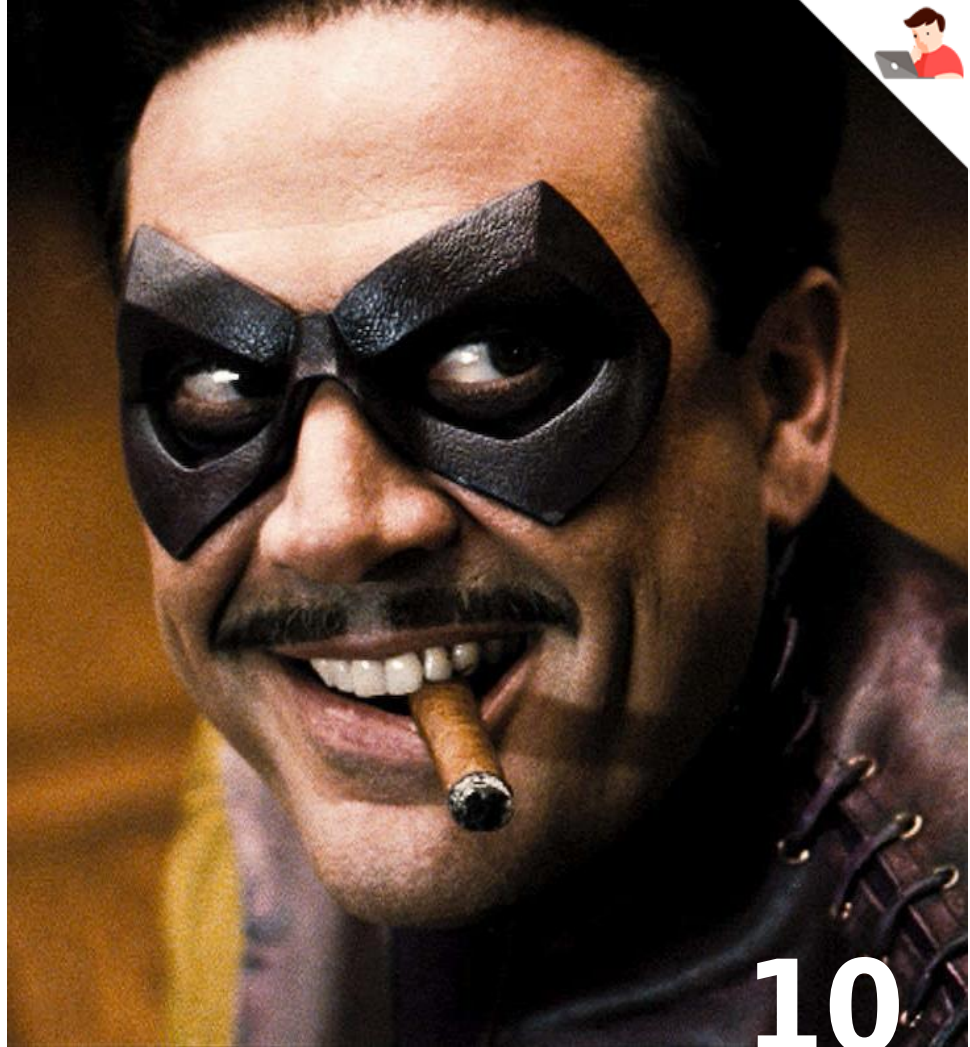
## Кроме того

Отключив таблицу стилей от вашего уже сверстанного сайта, можно легко прочитать и понять смысл сайта.

# Семантика очевидна

---

1. Заголовки `<h1>`
2. Параграфы `<p>`
3. Списки `<ul>`
4. Ссылки `<a>`
5. Таблицы `<table>`
6. Картинки `<img>`
7. Код `<code>`
8. Формы `<form>`, `<input>`





# Теги БЕЗ семантической нагрузки

---

## **<div>**

применяется для “оберток” любых отдельных блоков

## **<span>**

похож на div, но применяется к внутренним (инлайновым) элементам страницы, таким, как слова, фразы и т.п.



# Теги, которые мы НЕ используем

---

**<b>**

**<em>**

Результат их применения легко достигается при помощи CSS

# HTML5 теги

---

Повышаем **пуленепробиваемость** и **логичность** вёрстки.

Лишний открытый или закрытый div легко может поломать вёрстку. Но когда каркас сайта — атомарные и редко повторяющиеся html5-теги, то «поломка» локализуется в пределах html5-тега.



A low-poly, stylized illustration in shades of gray. On the left, a tank is partially visible, with its turret and tracks. To the right of the tank, a soldier stands in a dynamic, forward-leaning pose, possibly running or taking cover. The background is a simple, flat landscape with a horizon line. The overall style is minimalist and geometric.

# Войны значимости



# Стилизация по тегам и id

---

Это плохо.

К чему приводит?

- «войны значимости»
- скрытые проблемы (масштабируемость вёрстки и пр.).

С тегami ясно. Зачем тогда нужны id?

- Иногда используют для **секционных блоков** лендингов и пр.
- Для javascript (формы и пр.)
- Якоря для навигации
- Для примеров в учебниках



# Войны значимости

---

## Идентификатор

`id="имя"`

Уникален и должен быть включён лишь один раз

**Значимость - 100**

## Класс

`class="имя"`

может использоваться в коде неоднократно

**Значимость - 10**

## Тег

Стилизация тегов

нежелательна, ввиду низкой значимости и не уникальности

**Значимость - 1**





# Подведём итоги

---

- Добавляйте классы **ко всем тегам**, которые хотите стилизовать
- Используйте **методологию**, исключающую возможность пересечения стилей
- Не делайте слишком длинные цепочки (более 2х - 3х классов в цепочке):

`.header .logo .title a { color : red }`

- Однотипные элементы должны иметь одинаковые (общие) классы

## И ещё один совет

Разбивайте ваш CSS код по нескольким отдельным файлам для удобства его поддержки



# Display



# display: inline

---

**<span>**, **<a>**

- выстраиваются в строку
- интерпретируются как буквы
- аккуратно со свойствами **margin**, **padding**
- Уместны будут **line-height** и пр. свойства строк



# display: block

---

**<div>, <p>, <ul>, <li>**

Выстраиваются в столбец

К ним можно применять свойства width, height, margin, padding



# Почему float – это головная боль?

---

Все веб-разработчики знают, что делает свойство **Float**. Но не все правильно избавляются от **нежелательных последствий** применения этого свойства.

Когда внутри элемента содержатся другие элементы с **Float**, он схлопывается так, как будто в нем вообще нет дочерних элементов и его высота становится равна нулю. Зачастую это приводит к целому ряду нежелательных последствий.

Попытки решения этой проблемы зачастую не дают желаемого результата или приводят к новым неприятностям.

**clear: both**

или

**overflow: hidden**



# Свойство float и его корректная очистка

---

```
.clearfix:before,  
.clearfix:after {  
    content: " ";  
    display: table;  
}  
.clearfix:after {  
    clear: both;  
}
```

[Подробнее о том, как это работает](#)  
[Обзор прочих решений](#)



# display: inline-block

---

Один из наиболее популярных способов выстраивать блоки один за другим в обход float.

Но здесь есть подводные камни в виде **нежелательных пробелов**, которые возникают между элементами.





# Как «победить» пробелы?

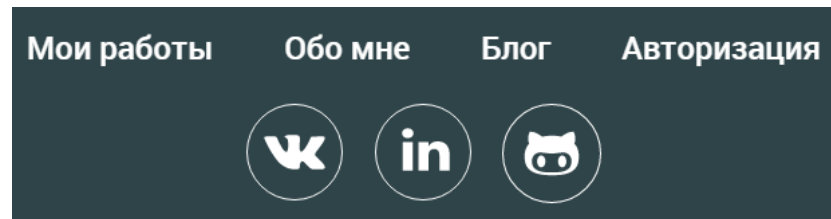
---

Мой любимый вариант, без «порчи»  
HTML кода

**font-size : 0px**

Где можно применить?

[Рассмотрим все варианты](#)  
[Кроссбраузерное решение](#)







# Сброс исходных настроек CSS

# Откуда растут ноги?

---

У всех браузеров есть «таблицы стилей **user agent**» — набор стилей CSS по умолчанию.

При этом, в каждом браузерном движке наборы по умолчанию немного отличаются.



# 4 способа

---

## Наиболее распространены

1. Антипаттерн, никогда так не делай!

```
*{margin: 0; padding: 0;}
```

2. CSS Reset Эрика Майера

3. Normalize.css Николаса Галлахера

4. border-box reset

**Twitter Bootstrap, HTML5 Boilerplate  
используют normalize.css.**

**А какой способ сброса применяете  
вы?**

# Meyer Reset

---

Сбрасываем **грубо**

Как установить?

**bower i -S reset-css**

**npm i -S reset-css**





# Normalize.css

---

Убирает **только различия** в браузерах, давая разработчикам **надежный фундамент** для вёрстки.

Как установить?

**bower i -S normalize.css**

**npm i -S normalize.css**




# Box-sizing reset

---

При помощи **box-sizing: border-box** мы говорим браузеру, что ширина, которую мы задаем, относится к элементу полностью, включая **border** и **padding**

```
html {  
  box-sizing: border-box;  
}
```

```
*, *:before, *:after {  
  box-sizing: inherit;  
}
```



# **Вертикальное и горизонтальное центрирование элементов**



# Горизонтальное центрирование

---

## Строчные элементы:

`text-align: center;`

## Блочные элементы:

`margin: 0 auto + фиксированная ширина`





# Вертикальное центрирование

---

**Чаще всего используются:**

1. Способ с line-height
2. Абсолютное позиционирование + отрицательный margin
3. transform: translate
4. Конечно, флексбоксы!

A low-poly, stylized illustration in shades of gray. On the left, a truck is shown from a side profile, with its cab and a large rectangular trailer. A cat is perched on top of the trailer, looking towards the right. The background features rolling hills and a horizon line. In the bottom left corner, there is a small red L-shaped graphic element. In the top right corner, there is a larger red L-shaped graphic element.

# Псевдоэлементы



# Псевдоэлементы

---

**<blockquote>**

**:: before**

Я так счастлив, мой друг, так  
упоен ощущением покоя, что  
искусство мое страдает от этого.  
Ни одного штриха не мог бы я  
сделать, а никогда не был таким  
большим художником, как в эти  
минуты.

**:: after**

**</blockquote>**

```
blockquote:before {  
  color: #bbb;  
  content: "\201C";  
}
```

```
blockquote:after {  
  color: #bbb;  
  content: "\201D";  
}
```

A low-poly, stylized illustration in shades of gray. On the left, a large truck is shown from a side profile. To its right, a person stands with their back to the viewer, looking towards the horizon. The background features rolling hills and a clear sky. Two red L-shaped corner brackets are present: one in the top right corner and another in the bottom left corner.

# Круглые картинки

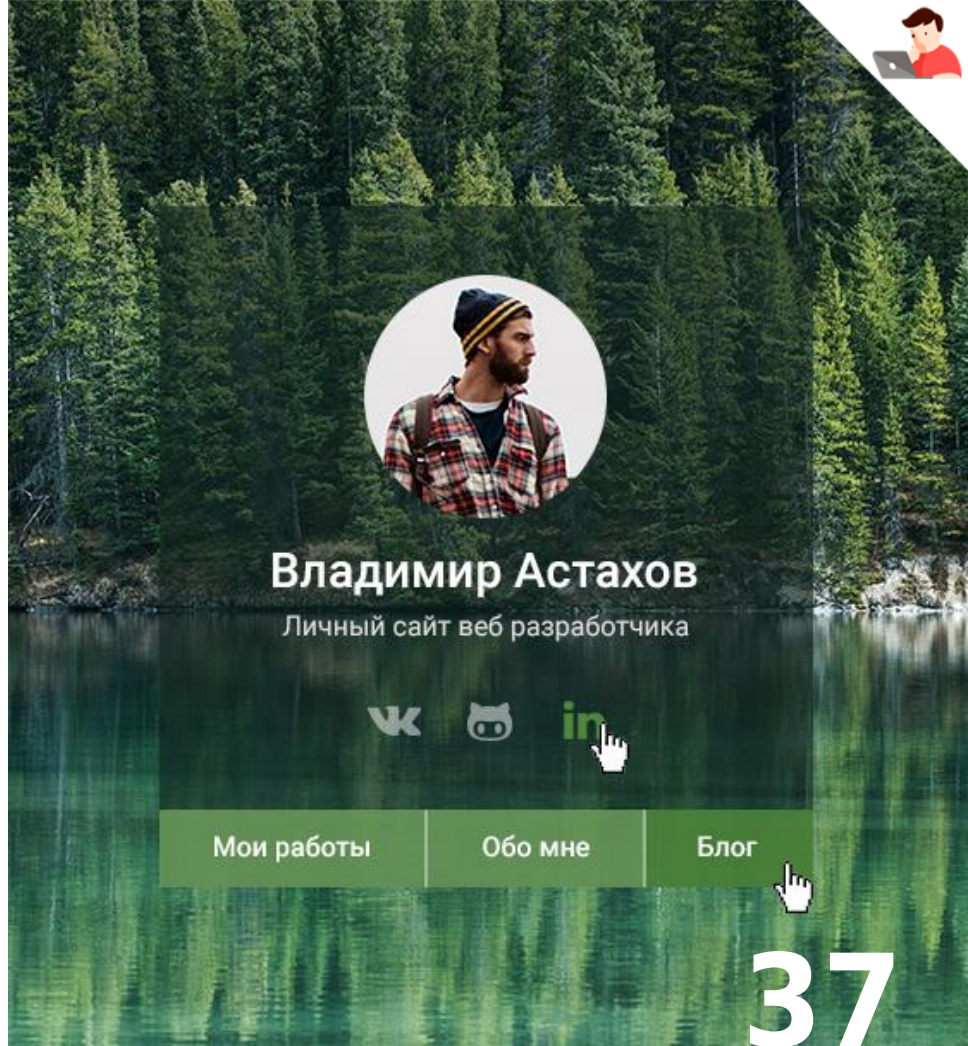
# Круглые картинки

---

Не бывает круглых картинок!

```
<div class="av-box"><img class="avatar"></div>
```

```
.av-box {  
  width: 150px;  
  height: 150px;  
  border-radius: 50% 50%;  
  overflow: hidden;  
}  
.avatar {  
  width: 100%;  
}
```



A low-poly, stylized illustration in shades of gray. On the left, a large truck is shown from a side profile, facing right. Behind the truck, a person is standing, looking towards the right. The background consists of simple, angular shapes representing a landscape or sky. In the top right corner, there is a red L-shaped line. In the bottom left corner, there is a red L-shaped line. The overall style is minimalist and geometric.

# Вопросы на засыпку



# Модель box

---

**Какое из следующих свойств не влияет на модель box?**

1. content
2. padding
3. margin
4. border
5. outline



# Модель box

---

**Какое из следующих свойств не влияет на модель box?**

5. outline ✓

*Обводка, созданная через свойство outline, рисуется «поверх» бокса, то есть outline всегда сверху, и не влияет на позицию и размер бокса или других боксов. Поэтому, показ или скрывание outline не приводит к пересчёту страницы.*





# Псевдо-классы

---

**Каков идеальный порядок  
следующих селекторов  
псевдо-классов?**

:link  
:visited  
:hover  
:active

# Псевдо-классы

---

Каков идеальный порядок  
следующих селекторов  
псевдо-классов?

Запомнить порядок можно при  
помощи мнемонического правила  
LVHA.

**Link → Visited → Hover → Active.**





# Перекрывтие элементов

---

**Какое из следующих свойств CSS, использованное само по себе, может вызвать перекрывтие элементов?**

1. z-index
2. margin
3. background
4. overflow



# Перекрывтие элементов

---

**Какое из следующих свойств CSS, использованное само по себе, может вызвать перекрывтие элементов?**

2. margin ✓

*Вам может показаться странным, что свойство, которое управляет отступами от края элемента, может привести к перекрывтию. Ответ прост: для этого необходимо использовать отрицательные значения.*



# Прячем элемент

---

**Какое свойство позволяет вам спрятать элемент, но сохранить занимаемое им пространство на странице?**

1. visibility
2. opacity
3. display
4. translate
5. overflow



# Прячем элемент

---

**Какое свойство позволяет вам спрятать элемент, но сохранить занимаемое им пространство на странице?**

1. visibility✓
2. opacity ✓

*Установив свойству visibility значение hidden, мы спрячем элемент. Он всё ещё будет занимать место на странице, равное его размерам. Также можно назначить ему opacity: 0.*



# Эффекты и псевдо-элементы

---

**Какие из следующих эффектов реализуется с помощью псевдо-элементов?**

1. Добавить к гиперссылке тень, которая появляется при наведении.
2. Изменить цвет элемента checkbox при его выборе.
3. Окрасить четные и нечетные строки таблицы различным цветом.
4. Всегда отображать жирным шрифтом первую строку абзаца.



# Эффекты и псевдо-элементы

---

**Какие из следующих эффектов реализуется с помощью псевдо-элементов?**

4. Всегда отображать жирным шрифтом первую строку абзаца ✓

*:first-line - единственный псевдо-элемент из перечисленного, остальное делается с помощью псевдо-классов.*





# line-height

---

**Если вы хотите установить двойной интервал на всем вашем сайте, какое из следующих значений line-height лучше всего подойдет для этой цели?**

1. 200%
2. 2em
3. 2
4. double



# line-height

---

**Если вы хотите установить двойной интервал на всем вашем сайте, какое из следующих значений line-height лучше всего подойдет для этой цели?**

2 ✓

*2 - указывает браузеру, что он должен сохранять соотношение размера шрифта и высоты строки  
double - такого свойства нет.*



# **.example.example.example**

---

**Какой цвет будет у текста элемента?**

`<div class="example another">Text</div>`

1. `.example {  
 color: red;  
}`

2. `.example.example {  
 color: green;  
}`

3. `.example.example.example {  
 color: orange;  
}`

4. `.example.another {  
 color: blue;  
}`



# .example.example.example

---

## Какой цвет будет у текста элемента?

```
<div class="example another">Text</div>
```

3. ✓

```
.example.example.example {  
  color: orange;  
}
```

*3. Перечисление одного и того же класса добавляет вес к селектору. Так можно увеличить вес селектора не делая его контекстно зависимым.*



# **.example.example.example**

---

**Какая высота строки будет у текста в элементе с классом example?**

```
.example {  
  font-size: 20px;  
  line-height: 1.5;  
}
```

- 1. 20px**
- 2. 40px**
- 3. 30px**
- 4. 35px**



# `.example.example.example`

---

**Какая высота строки будет у текста в элементе с классом `example`?**

```
.example {  
  font-size: 20px;  
  line-height: 1.5;  
}
```

4. 35px ✓

$$20 * 1.5 = 35$$