6.6. Управление соотношением сторон замещаемых элементов

~ 9 минут

В идеальном мире верстальщик берёт картинку из макета, вставляет её в вёрстку и получает безупречный результат. Но в жизни всё работает иначе, и картинки могут приходить от контент-менеджера в любых разрешениях и пропорциях. Задача верстальщика — подготовиться к этому и показать лучшее возможное отображение картинки. В этом поможет `object-fit`.

`object-fit` определяет, как элемент реагирует на размеры своего бокса. Он предназначен для замещаемых элементов — `<img>`, `<video>`, `<object>`, `<input type="image">`. Все примеры мы рассмотрим на теге `<img>`, но они будут актуальными и для других замещаемых элементов.

`object-fit` позволяет обрезать встроенное изображение, предоставляя контроль над тем, как оно будет сжиматься и растягиваться.

Каждое изображение имеет оригинальный размер и окно отображения — бокс. Например, картинка может иметь размеры `1920x1080`, а бокс будет `1024x768`. Изображение сожмётся до бокса и мы будем видеть сжатые размеры, но при этом оригинальный размер картинки не изменится.

`object-fit` можно задать одно из этих пяти значений:

— `fill`: значение по умолчанию, которое растягивает изображение до размеров бокса, независимо от соотношения его сторон.

— `contain`: увеличивает или уменьшает размер изображения, чтобы заполнить бокс, сохраняя соотношение его сторон.

— `cover`: изображение будет заполнять высоту и ширину бокса, сохраняя соотн сторон, но это значение часто обрезает изображение.

— `none`: изображение проигнорирует высоту и ширину бокса и сохранит исходный размер.

— `scale-down`: изображение сравнивает разницу между `none` и `contain` для определения наименьшего размера.

Демонстрации

Оригинал

Оригинальное изображение имеет пропорции примерно 3:2.

```
<img class="image" src="original.png" width="300" height="193" alt="Кексик">
```

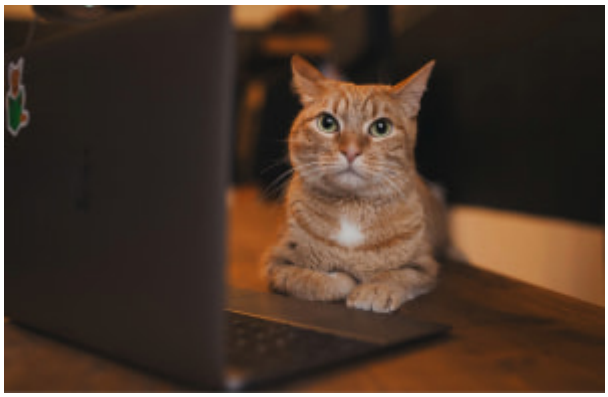


Фото Кекса в оригинальном размере

Теперь увеличим размер бокса по ширине, чтобы увидеть разницу в значениях `object-fit`, и добавим фоновый цвет, чтобы видеть размер бокса. Если вы не видите серого фона под картинкой, значит картинка заняла весь бокс.

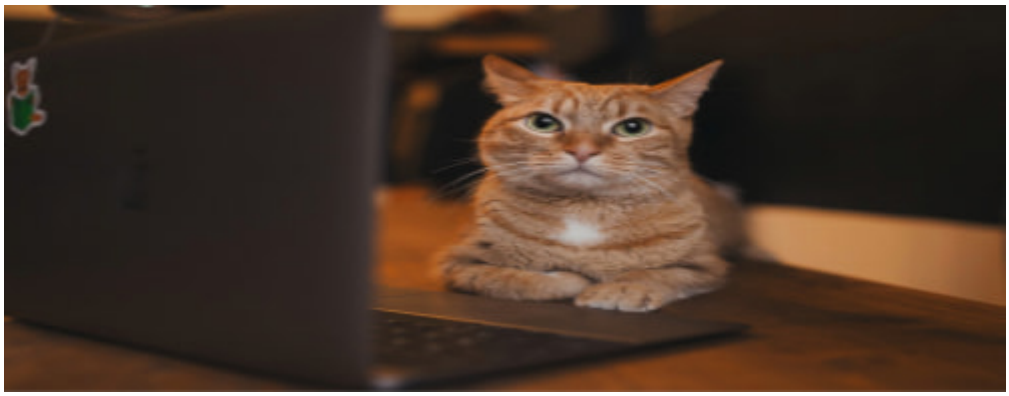
```
.image {
  width: 500px;
  background-color: #ccc;
}
```

`fill`

Задача `fill` — растянуть картинку до размеров её бокса. Из-за этого могут получаться сплющенные картинки.

```
<img class="image" src="original.png" width="300" height="193" alt="Кексик">
```

```css
.image {
  width: 500px;
  background-color: #ccc;
  object-fit: fill;
}
```



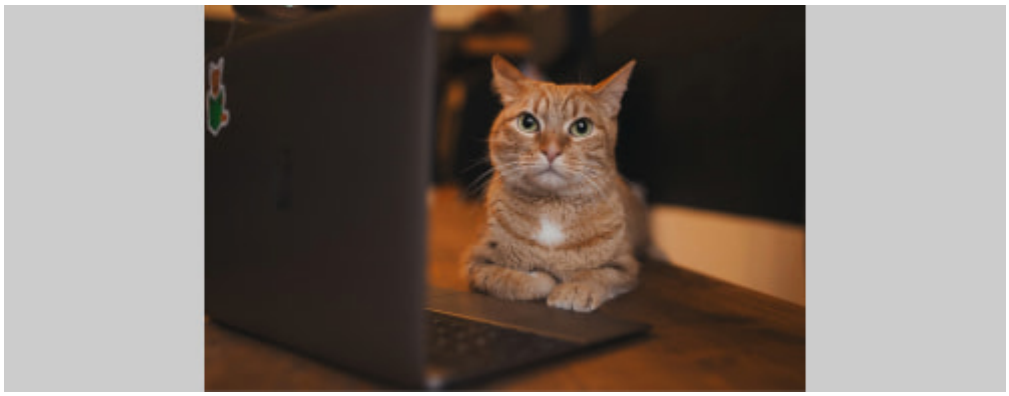
Свойство fill растянуло фотографию

`contain`

Задача `contain` — сохранить пропорции оригинальной картинки. При этом бокс заполнять не обязательно.

```html
<img class="image" src="original.png" width="300" height="193" alt="Кексик">
```

```css
.image {
  width: 500px;
  background-color: #ccc;
  object-fit: contain;
}
```



Пропорции фотографии сохранились

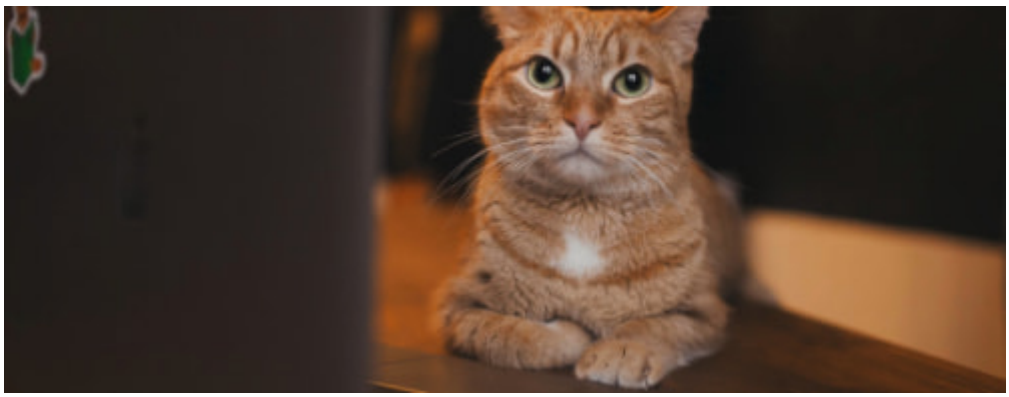
Серым цветом показан настоящий размер бокса. Картинка не исказилась и сохр
пропорции оригинала.

cover

`cover` растянет картинку по всему боксу, но при этом сохранит пропорции картинки и обрежет всё, что не поместилось. Обратите внимание, сколько места от верхнего края картинки до ушей Кекса в оригинальной картинке и сколько здесь. Уши пришлось подрезать, чтобы сохранились пропорции оригинальной картинки и она не искажалась как с `fill`.

```
<img class="image" src="original.png" width="300" height="193" alt="Кексик">
```

```
.image {
  width: 500px;
  background-color: #ccc;
  object-fit: cover;
}
```



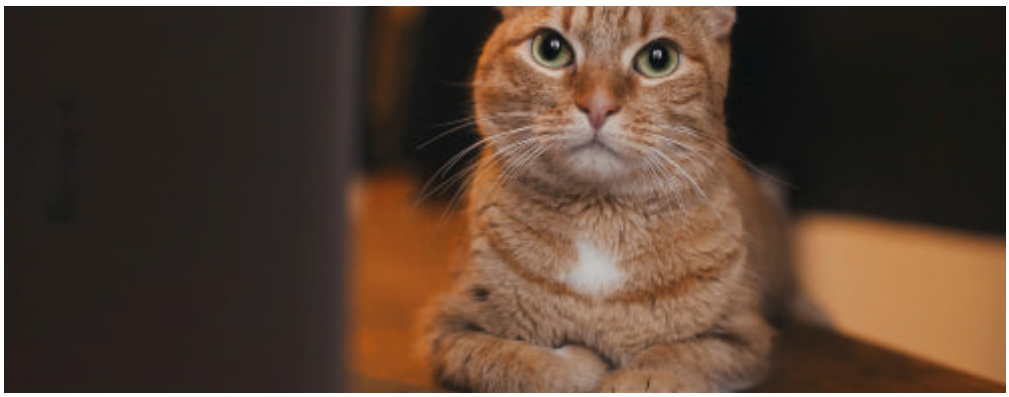
Уши Кекса остались за кадром

none

Изображение проигнорирует высоту и ширину бокса и сохранит исходный размер.

```
<img class="image" src="original.png" width="300" height="193" alt="Кексик">
```

```
.image {
  width: 500px;
  background-color: #ccc;
  object-fit: none;
}
```



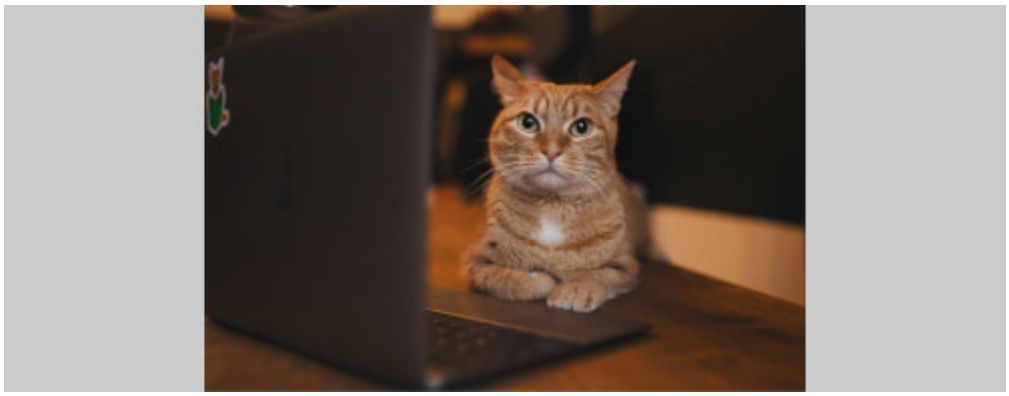
Часть фотографии оказалась отрезанной

`scale-down`

Задача `scale-down` — сравнить разницу между `none` и `contain`, определить наименьший размер и вставить тот, который меньше: `none` или `contain`.

```
<img class="image" src="original.png" width="300" height="193" alt="Кексик">
```

```
.image {
  width: 500px;
  background-color: #ccc;
  object-fit: scale-down;
}
```



Свойство scale-down впишет изображение

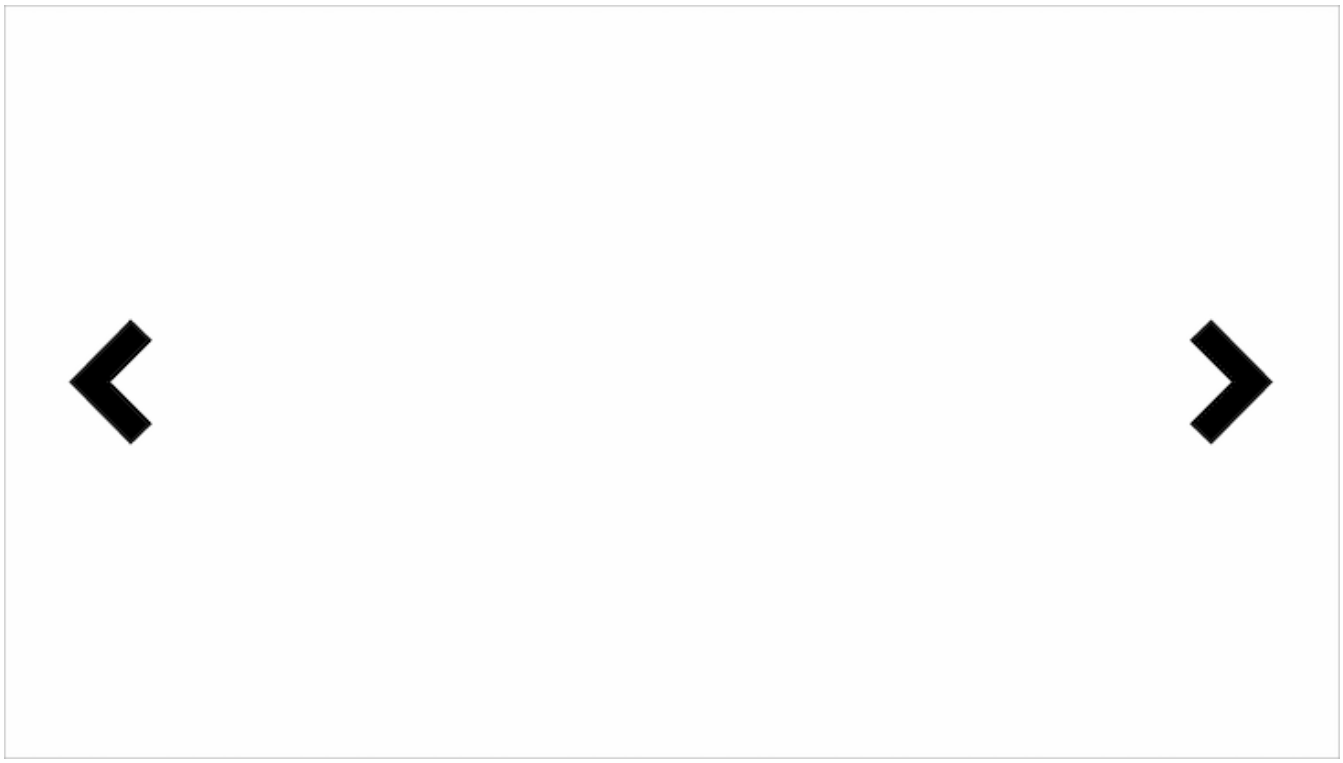
Итог

В реальных проектах чаще всего используются значения `contain` и `cover`, так как эти значения сохраняют пропорции, не искажают картинки и уже знакомы верстальщикам по свойству `background-size`.

Методика использования

Галерея

Представим, что вы сверстали галерею по макету. Схематично она выглядит так:



Пустая галерея

Вы экспортируете графику и вставляете в галерею.

Галерея с графикой

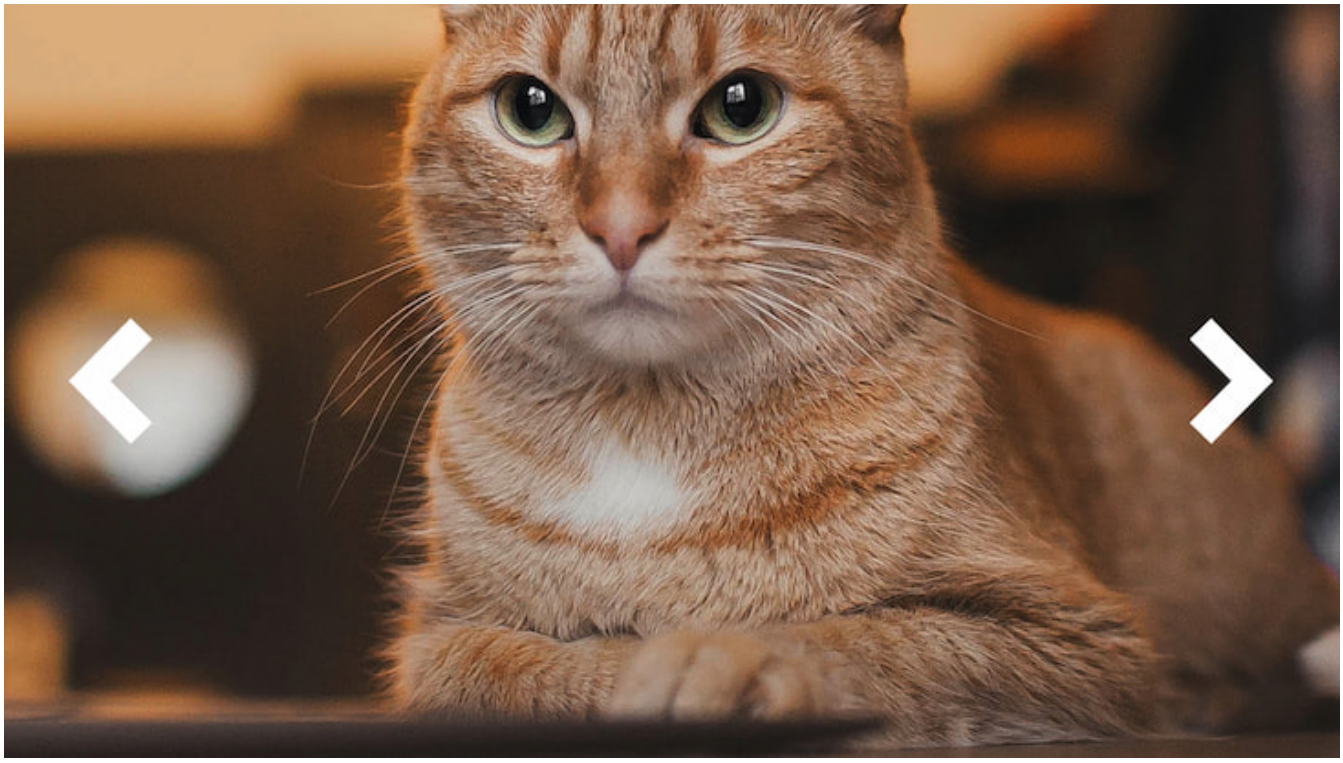
Идеальная вёрстка: пропорции галереи и графики совпадают — 16:9, ничего не искажается и не обрезается. Работа закончена, можно сдавать её заказчику.

Проблемы начнутся, когда к работе приступит контент-менеджер. Он сфотографирует своего любимого кота и вставит фотографию без обработки в вашу идеальную галерею.

Вёрстка растягивает изображение

Мы уже знаем, что исправить ситуацию можно при помощи `object-fit`. Но какое значение использовать?

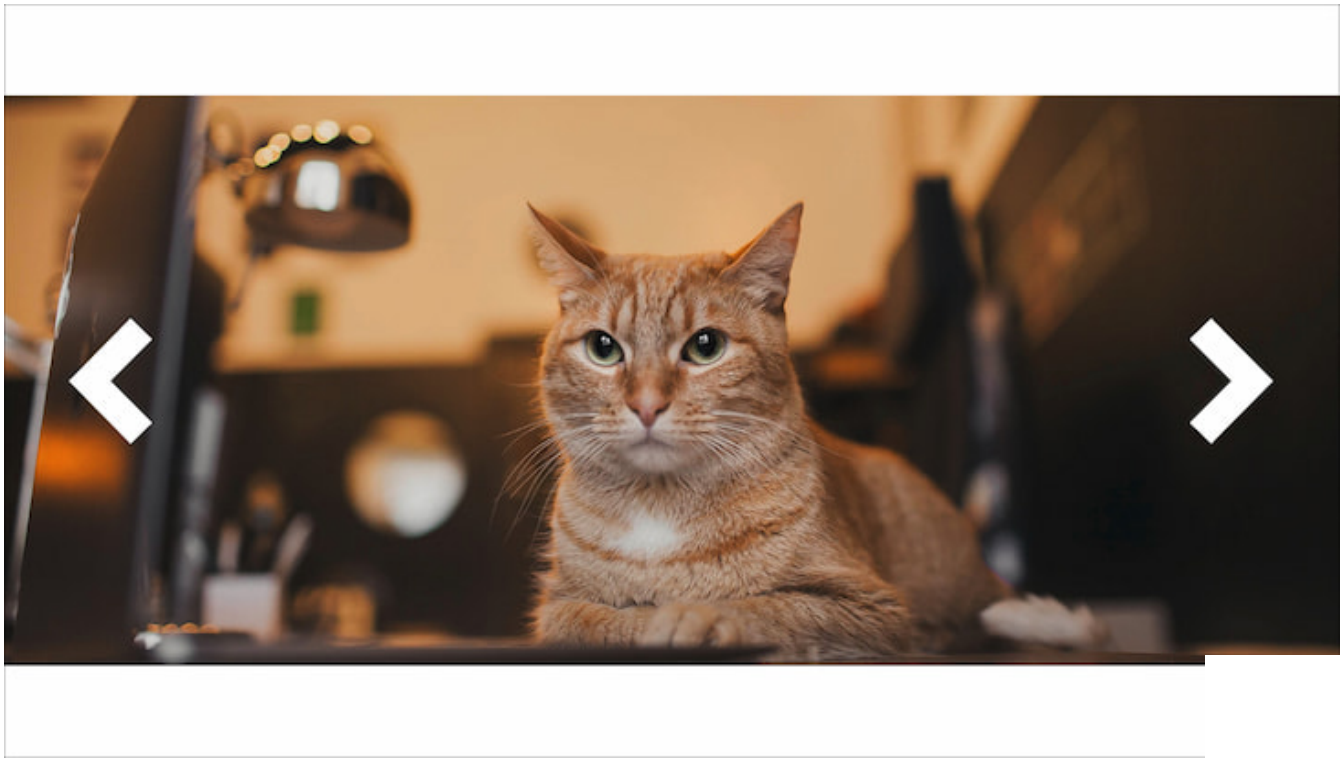
`cover` может обрезать важные детали фотографии.



Кадрирование фотографии при использовании cover

Кекс остался без ушей, такой вариант нам не подходит. Наша задача — показать всё изображение и при этом не исказить его, а `cover` может срезать важные детали фотографии. Представьте себе рекламу, в которой демонстрируют крем для лица. На изображении человек держит крем в руках, но `cover` скрыл руки вместе с кремом за боксом. Это недопустимо.

`contain` решает нашу задачу — показать всё изображение и не исказить его.



Горизонтальное изображение с полями

У фотографии появились поля, но это лучше, чем искажённое изображение. Более того, такой способ хорошо подходит и для вертикальных фотографий.

Вертикальное изображение с полями

Вывод:

Если у элемента, как у нашей галереи, жёстко зафиксированы размеры и нет возможности контролировать, какие изображения будут добавляться в вёрстку, нужно сохранять пропорции картинки, чтобы избежать её искажения. Чаще всего для этой цели подходит `object-fit: contain`.

Контент

В контентной части тоже нужно соблюдать правило сохранения пропорций без искажений.

«МОЛОДЁЖНЫЙ» КВИФФ

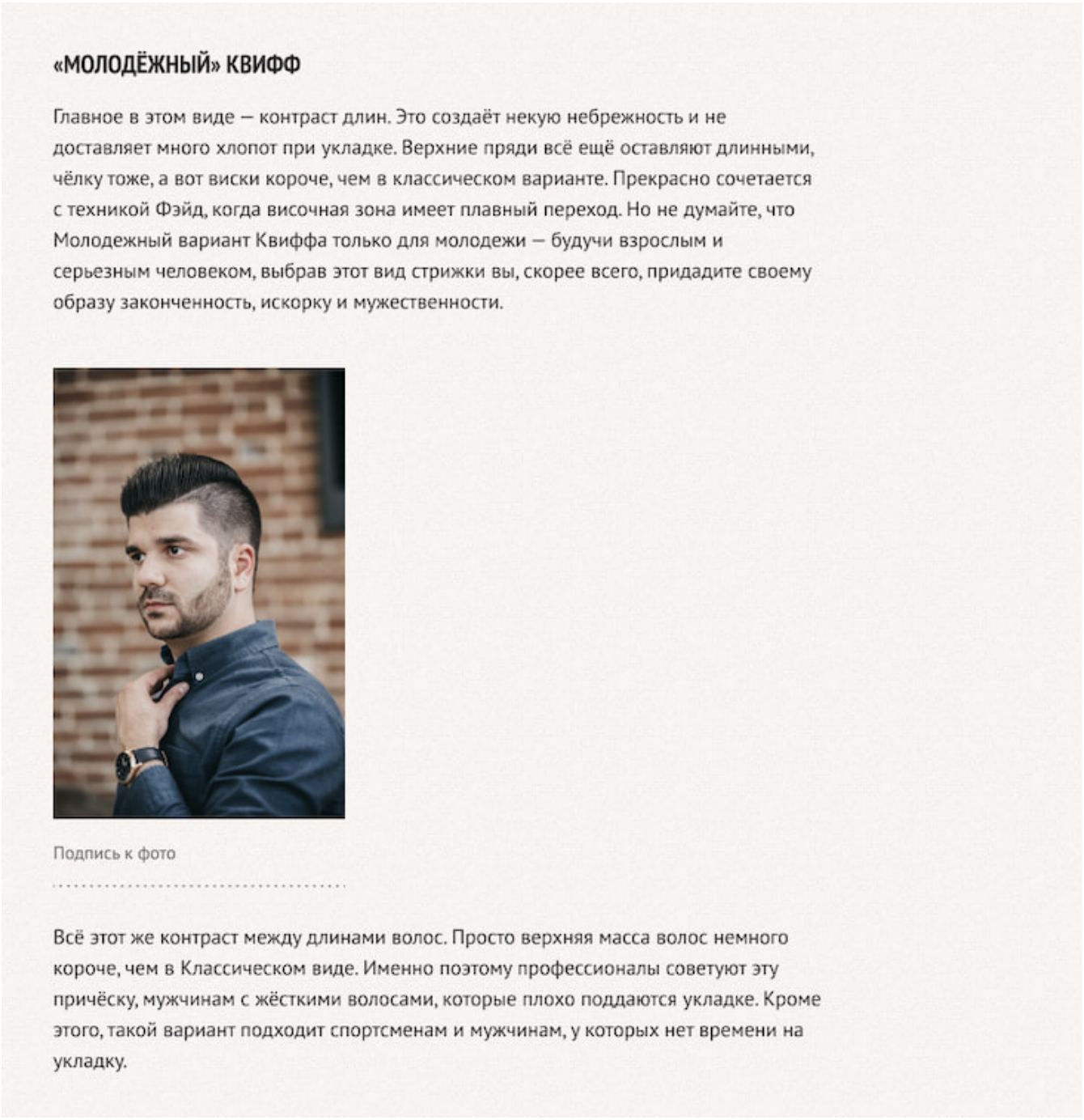
Главное в этом виде — контраст длин. Это создаёт некую небрежность и не доставляет много хлопот при укладке. Верхние пряди всё ещё оставляют длинными, чёлку тоже, а вот виски короче, чем в классическом варианте. Прекрасно сочетается с техникой Фэйд, когда височная зона имеет плавный переход. Но не думайте, что Молодежный вариант Квиффа только для молодежи — будучи взрослым и серьезным человеком, выбрав этот вид стрижки вы, скорее всего, придадите своему образу законченность, искорку и мужественности.

Подпись к фото

Всё этот же контраст между длинами волос. Просто верхняя масса волос немного короче, чем в Классическом виде. Именно поэтому профессионалы советуют эту причёску, мужчинам с жёсткими волосами, которые плохо поддаются укладке. Кроме этого, такой вариант подходит спортсменам и мужчинам, у которых нет времени на укладку.

Фотография внутри контента

Например, если картинка изменится, то контент просто перестроится, и ничего дополнительного для сохранения пропорций делать не нужно.



Небольшая фотография не ломает вёрстку

Вывод:

Если контент вокруг картинки перестраивается в зависимости от размеров картинки, то для сохранения пропорций ничего делать не нужно.

Обратите внимание: если добавить в разметку огромную картинку, то она может сломать сетку. Чтобы защититься от таких ситуаций, нужно добавить для изображений `max-width: 100%`. Тогда максимальный размер картинки всегда будет соответствовать размеру её родителя, и она не сможет выйти за его пределы. Этому правилу стоит следовать всегда.

Логотипы

В прошлых примерах мы рассмотрели изображения, чьи пропорции допустимо немного изменять. Но есть и такие изображения, которые запрещено искажать или обрезать — это логотипы компаний. Компании вкладывают миллионы в свои логотипы и точно знают, как они должны выглядеть.

Логотипы партнёров

Мы не знаем, какие партнёры в будущем могут быть у заказчика. У одного из партнёров может оказаться вертикальный логотип.

В списке появился новый логотип

Для ситуаций, когда ограничена область показа и когда мы точно знаем, что изображение не должно изменяться, используйте `object-fit: contain`.

Логотип уменьшился, но сохранил пропорции

Видно, что логотип Levi's получается маленьким и вокруг него образуется много пустого пространства. Но это лучше, чем искажённый логотип.

Вывод:

Логотипы никогда не должны искажаться.

Прочитали главу?

Нажмите кнопку «Готово», чтобы сохранить прогресс.

Готово

Если вы обнаружили ошибку или неработающую ссылку, выделите ее и нажмите Ctrl + Enter

Поиск по материалам

Git

Все материалы

В самом начале

- [ ] Пройдите опрос
- [ ] Укажите персональные данные
- [ ] Изучите регламент
- [ ] Прочитайте FAQ
- [ ] Добавьте свой Гитхаб
- [ ] Выберите наставника
- [ ] Создайте проект

Мой наставник

Выбрать наставника

Работа с наставником

У вас осталось **10** из 10 консультаций.

История




Практикум

Тренажёры

Подписка

Для команд и компаний

Учебник по PHP

Профессии

Фронтенд-разработчик

JavaScript-разработчик

Фулстек-разработчик

Курсы

HTML и CSS. Профессиональная вёрстка сайтов

HTML и CSS. Адаптивная вёрстка и автоматизация

JavaScript. Профессиональная разработка веб-интерфейсов

JavaScript. Архитектура клиентских приложений

React. Разработка сложных клиентских приложений

Node.js. Профессиональная разработка REST API

Node.js и Nest.js. Микросервисная архитектура

TypeScript. Теория типов

Алгоритмы и структуры данных

Паттерны проектирования

Webpack

Vue.js 3. Разработка клиентских приложений

Git и GitHub

Анимация для фронтендеров

Блог

С чего начать

Шпаргалки для разработчиков

Отчеты о курсах

Услуги

Работа наставником

Для учителей

Стать автором

Информация

Об Академии

О центре карьеры

Остальное

Написать нам

Мероприятия

Форум

Соглашение

Конфиденциальность

Сведения об образовательной организации

Лицензия № 4696

Sk
Участник