

Вёрстка многослойных элементов интерфейса

Выполнен на 51%

Введение

Углублённая теория

Позиционирование

Как работает position: sticky

z-index

Многослойность с помощью CSS Grid Layout

Про vh, vw и другие единицы измерения

Словарь терминов

Методика вёрстки многослойных элементов

Кейс 1, лёгкий уровень

Кейс 2, лёгкий уровень

Кейс 3, лёгкий уровень

Кейс 4, лёгкий уровень

Кейс 5, средний уровень

Кейс 6, средний уровень

Кейс 7, средний уровень

Кейс 8, сложный уровень

Кейс 9, сложный уровень

Кейс 10, сложный уровень

Главная / Моё обучение / Вёрстка многослойных элементов интерфейса / Углублённая теория /

Про vh, vw и другие единицы измерения

`vw`, `vh`, `vmIn`, `vmax` — это единицы измерения относительно экрана. Они поддерживаются всеми современными браузерами.

— `vw` — 1% ширины окна;

— `vh` — 1% высоты окна.

В отличие от `%`, единицы измерения `vw` и `vh` не требуют установки значений по цепочке родительских элементов, так как их значение вычисляется напрямую относительно окна браузера. И величина `100vh` всегда составляет всю высоту окна браузера, а `100vw` — всю ширину окна браузера.

Наглядно это демонстрирует пример ниже. У нас есть контейнер шириной `300px`. В этот блок добавим два блока, первый шириной `100%`, второй — `100vw`.

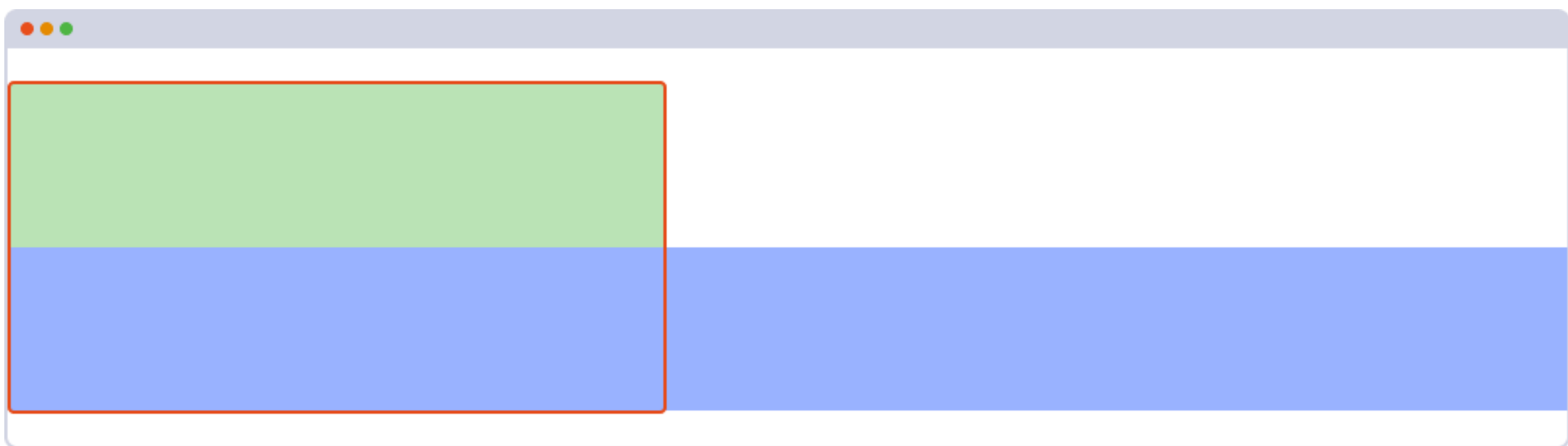
```
<div class="block">
  <div class="block-item block-item-percent">
  </div>
  <div class="block-item block-item-vw">
  </div>
</div>
```

```
.block {
  width: 300px;
  height: 200px;
  background-color: #aa0000;
  outline: 2px solid #aa0000;
}

.block-item {
  height: 100px;
}

.block-item-percent {
  width: 100%;
  background-color: #00aa00;
}

.block-item-vw {
  width: 100vw;
  background-color: #0000aa;
}
```



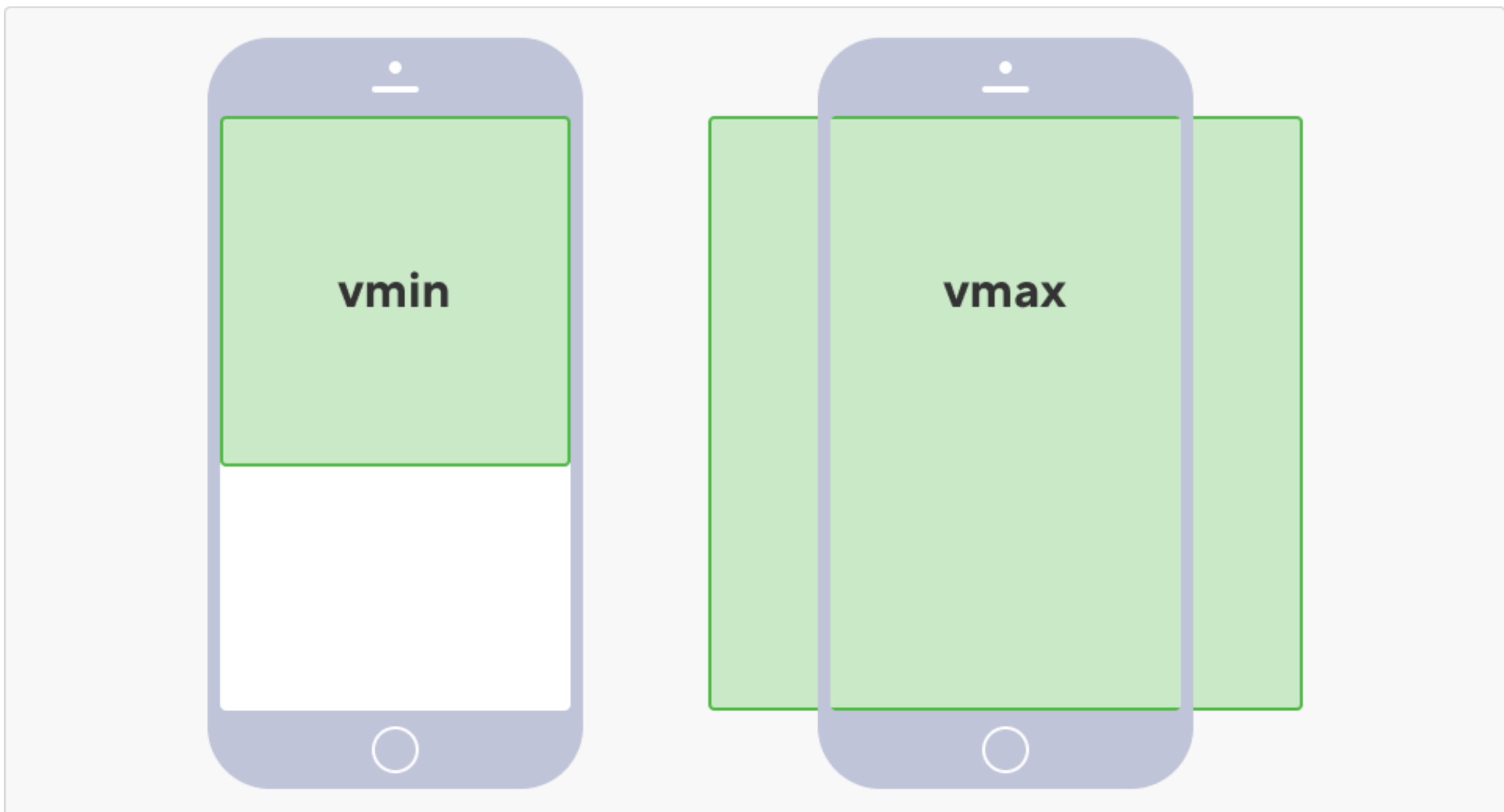
Блок с шириной равной `100%` и блок с шириной `100vw`

— `1vmIn` — меньшее из значений `1vw` и `1vh`;

— `1vmax` — большее из значений `1vw` и `1vh`.

Также имеет, если ширина окна браузера равна `1200px`, а высота `700px`, то `1vmIn` будет равен `7px`, а `1vmax` — `12px`.

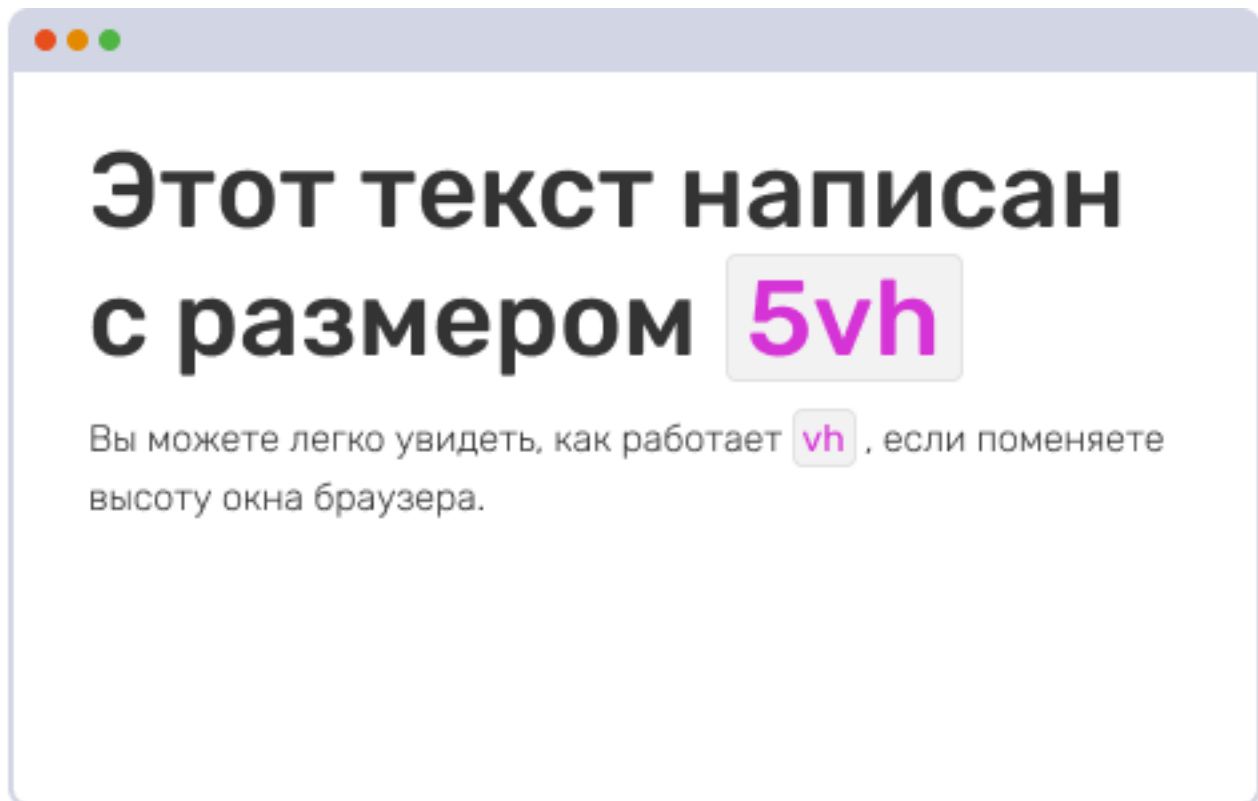
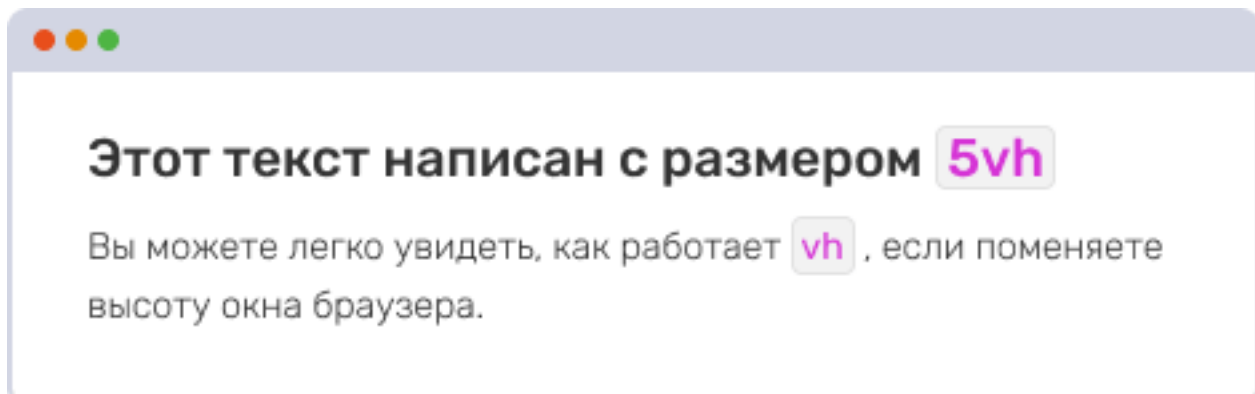
Как эти единицы измерения выглядят на экране мобильного телефона.



`100vmIn` и `100vmax` на экране мобильного телефона

`vw`, `vh`, `vmIn`, `vmax` очень полезны для адаптивных сайтов и сайтов, которые должны работать на мобильных устройствах: вне зависимости от параметров экрана элемент подстроится под размеры выюпорта, который есть в наличии.

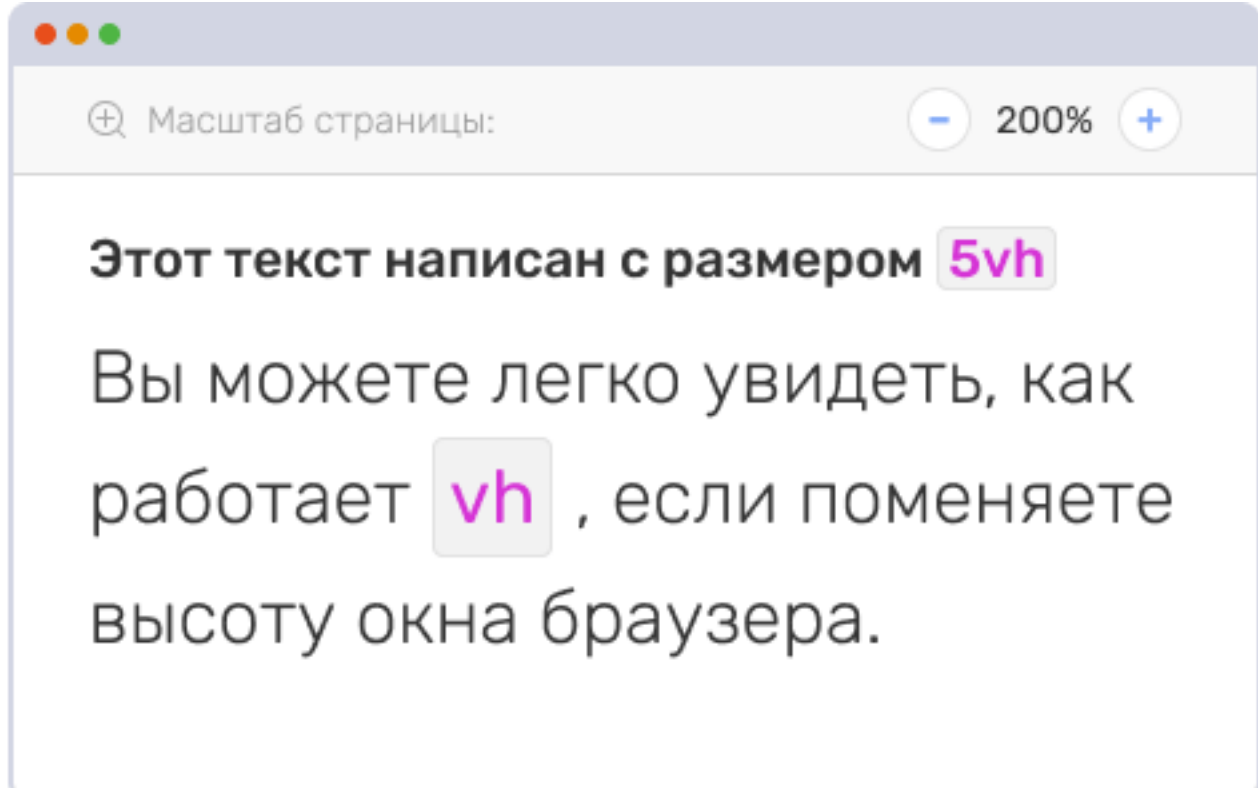
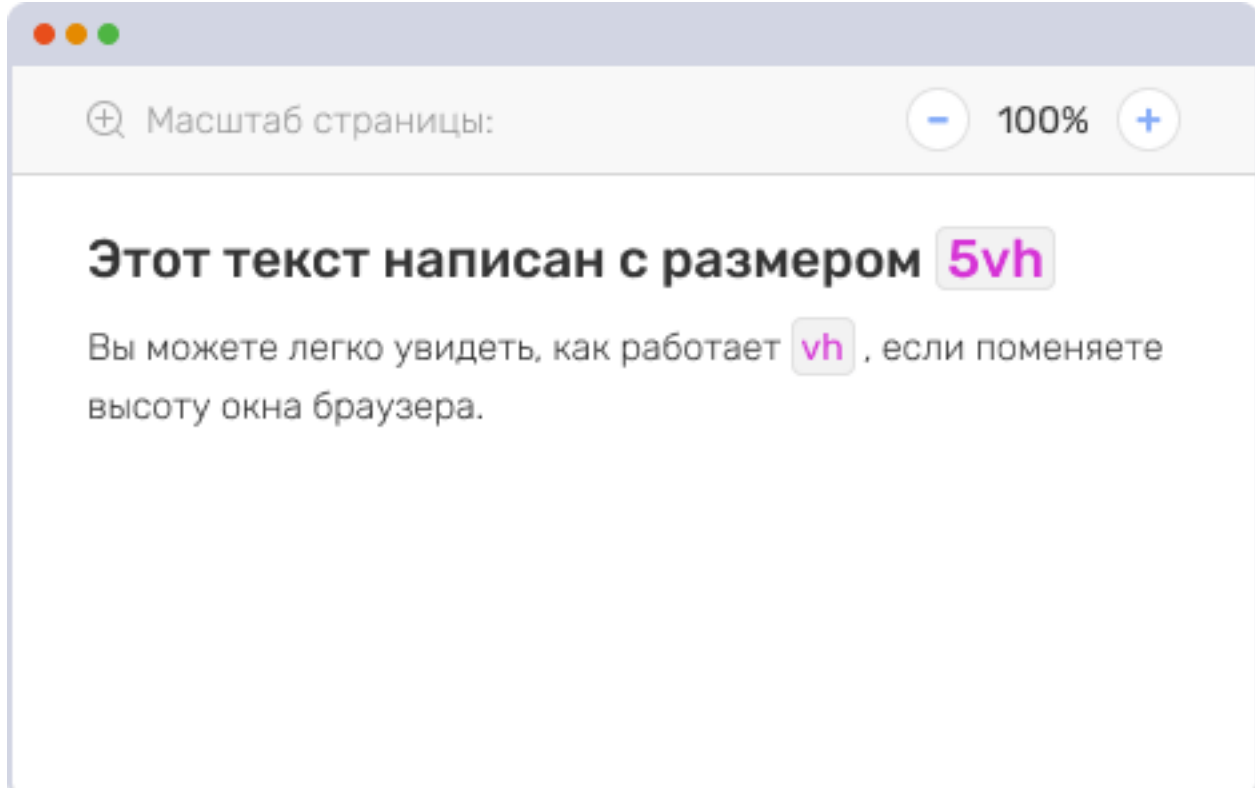
В примере ниже размер основного текста — `12px`, для выделенного текста установлена высота `5vh`. Посмотрите, как меняется размер элемента при увеличении высоты окна браузера.



Тест размером `5vh` при разной высоте окна браузера

Вот ещё один пример. Посмотрите, как ведёт себя текст размером `5vh` при изменении масштаба страницы.

Основной текст всё также имеет размер `12px`.



Тест размером `5vh` при масштабе страницы `100%` и `200%`

Замечание про появление горизонтального скrolла при использовании `100vw` в десктопной версии сайта

Если задать элементу ширину `100%`, получится ширина родителя, который, скорее всего, занимает только часть экрана, а `100vw` позволяет растянуть именно на ширину окна браузера не обращая внимания на размеры родительских элементов. Из-за этой особенности следует учитывать, что `100vw` — это ширина всего окна вместе с полосой прокрутки. Доступная для контента ширина окна полосу прокрутки не включает, из-за чего при попытке задать элементам ширину выюпорта появится горизонтальный скролл.

Проблема с полосами прокрутки будет видна только в десктопных браузерах, потому что на мобильных скроллбар размещается поверх страницы и не занимает пространство.

Многослойность с помощью CSS Grid Layout

Словарь терминов



Практикум

Тренажёры

Подписка

Для команд и компаний

Учебник по РНР

Профессии

Фронтенд-разработчик

JavaScript-разработчик

Фулстек-разработчик

Услуги

Работа наставником

Для учителей

Стать автором

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

Информация

Об Академии

О центре карьеры

Остальное

Написать нам

Мероприятия

Форум