

[Начать с нуля](#)[Получить профессию](#)[Прокачать специализацию](#)[\*\*<!доктайп>\*\*](#)[HTML](#)[CSS](#)[JS](#)[Git](#)[Софт](#)[Айти](#)[Случайная статья](#)

# Как отобразить элемент на странице. Свойство display

13 мая 2023

[CSS](#)

Саша Ермайкина

Свойство `display` определяет тип отображения элемента, как он будет отображаться на веб-странице: как блок, строчный элемент или каким-то другим способом.

## Значения свойства `display`

- `block`,
- `inline`,
- `inline-block`,
- `none`.

Теперь разберём на примерах значения свойства `display`.

`block` – элемент отображается как блок и занимает всю доступную ширину. Новый блок начинается с новой строки. Примером элемента, который имеет значение `display: block`, может быть [`div`](#).

```
.block {  
    display: block;  
    background-color: #ccc;  
    width: 500px;  
    height: 300px;  
}  
  
<div class="block">  
    <h2>Заголовок блока</h2>  
    <p>Текст внутри блока</p>  
</div>
```



**Канал для будущих мидлов**

# Заголовок блока

Текст внутри блока

`inline` – элемент отображается как строчный элемент и не создаёт новый блок. Элемент занимает столько места, сколько ему нужно для отображения содержимого. Примером элемента, который имеет значение `display: inline`, может быть [span](#).

<p>Этот текст <span class="inline">выделенный текст</span> и еще немного текста</p>

```
.inline {  
    display: inline;  
    background-color: yellow;  
    padding: 5px;  
}
```

Этот текст выделенный текст и еще немного текста

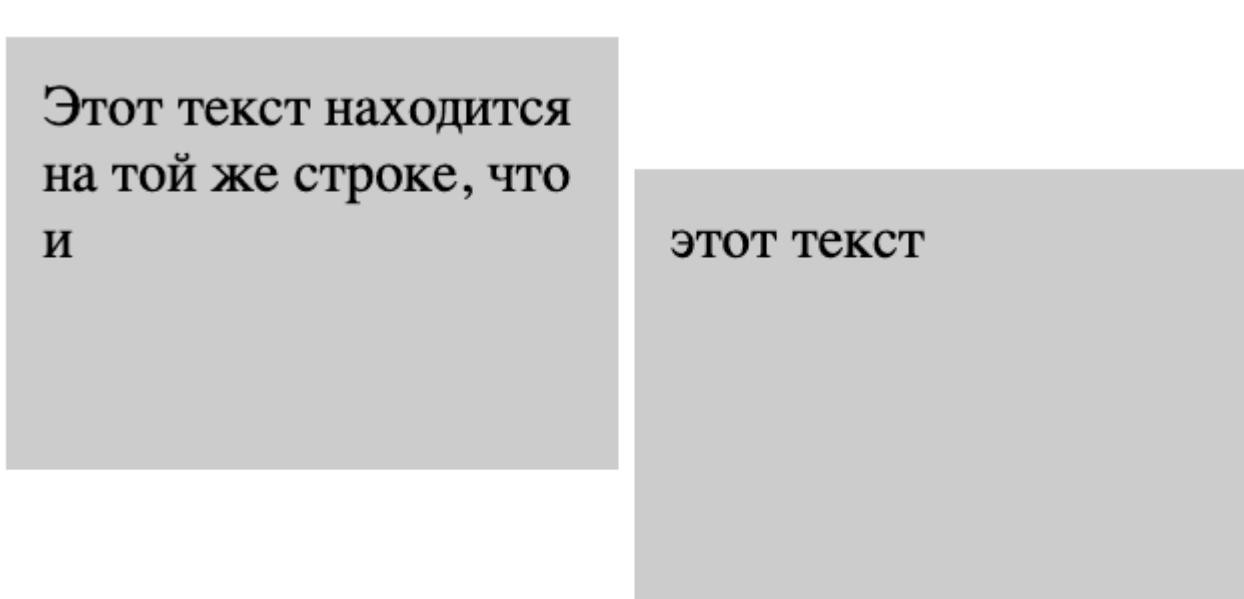
`inline-block` – элемент отображается как строчный элемент, но его содержимое может быть отформатировано как блок. Элемент занимает столько места, сколько ему нужно для отображения содержимого, но может иметь ширину и высоту. Примером элемента, который имеет значение `display: inline-block`, может быть [button](#).



Канал для  
будущих мидлов

```
<div class="inline-block">Этот текст находится на той же строке, что и</div>
<div class="inline-block">этот текст</div>
```

```
.inline-block {
    display: inline-block;
    background-color: #ccc;
    width: 150px;
    height: 100px;
    padding: 10px;
}
```



none – элемент не отображается на веб-странице и его содержимое не занимает места. Это может быть полезно для скрытия элементов на странице или для создания анимаций. Примером элемента, который имеет значение display: none, может быть script.



**Канал для будущих мидлов**

```
.hidden {  
    display: none;  
}  
  
<div class="hidden">Этот элемент скрыт</div>
```

# Вёрстка — база для фронтенданда

Научитесь создавать сайты и веб-приложения на профессии «Фронтенд-разработчик». После обучения — стажировка, портфолио, резюме и гарантия трудоустройства.

Записаться

Пример использования свойства `display` разными значениями.

```
<div class="block-example">Этот блочный элемент занимает всю ширину.</div>
```

```
<span class="inline-example">Этот строчный элемент занимает только столько места, сколько ему нужно.</span>
```

```
<button class="inline-block-example">Этот элемент имеет значение inline-block, поэтому его содержимое может быть отформатировано как блок и иметь ширину и высоту.</button>
```

```
<div class="hidden-element">Этот элемент имеет значение none, поэтому он не отображается на веб-странице.</div>
```

Стилизуем элементы так:



Канал для будущих мидлов

```
.block-example {
  display: block;
  background-color: yellow;
}

.inline-example {
  display: inline;
  background-color: pink;
}

.inline-block-example {
  display: inline-block;
  background-color: lightblue;
  width: 200px;
  height: 50px;
}

.hidden-element {
  display: none;
}
```

Этот блочный элемент занимает всю ширину.

Этот строчный элемент занимает только столько места, сколько ему нужно.

Этот элемент имеет значение `inline-block`, поэтому его содержимое может быть отформатировано как блок и иметь ширину и высоту.

Блочный элемент имеет жёлтый фон, строчный элемент – розовый фон, элемент `button` – голубой фон и имеет ширину и высоту, а скрытый элемент не отображается на странице.

## Наследуется ли свойство `display`

Свойство `display` является не наследуемым, что означает, что тип отображения элемента не будет наследоваться его потомками. Тип отображения элемента должен быть указан явно для каждого элемента на странице.

Например, если родительский элемент имеет свойство `display: block;`, а его дочерний элемент не имеет заданного свойства `display`, то тип отображения дочернего элемента будет определён по умолчанию от типа элемента.

Однако если вы зададите



**Канал для будущих мидлов**

то оно

переопределит наследование значения и будет применяться только к этому элементу.

```
<div class="parent">
  <p>Это абзац, наследующий свойства родительского элемента</p>
  <div class="child">
    <p>Это абзац с явно заданным типом отображения</p>
  </div>
</div>
```

Стили для блоков:

```
.parent {
  display: block;
}

.child {
  display: inline;
}
```

Здесь родительский элемент имеет свойство `display: block;`, но дочерний элемент `.child` имеет свойство `display: inline;`. Поэтому абзац, находящийся в дочернем элементе, будет отображаться как инлайновый элемент, несмотря на то, что родительский элемент имеет тип отображения блочного элемента.

## Нюансы использования

Как и любое CSS-свойство, `display` имеет нюансы и особенности, которые важно учитывать при создании веб-страниц.

- `display` определяет, как элемент будет взаимодействовать с другими элементами на странице. Например, элементы с типом `block` занимают всю доступную ширину и начинаются с новой строки, что может повлиять на позиционирование других элементов на странице.
- Некоторые элементы имеют собственные значения по умолчанию для свойства `display`. Например, `div` имеет значение по умолчанию `display: block`, а `span` имеет значение `display: inline`. Поэтому, если вы хотите изменить значение `display` для этих элементов, необходимо
- При использовании зн



**Канал для будущих мидлов**

итывать, что между  
ан пробелом в HTML-

коде. Этот нюанс можно устраниить разными способами, например, удалением пробелов в коде или применением стилей, которые устраняют эффект.

★ Поддержка браузерами свойства [display](#)

## Материалы по теме

- [CSS-свойство position](#)
- [Как прятать](#)
- [CSS-свойство max-height](#)

«Доктайп» – журнал о фронтенде. Читайте, слушайте и учитесь с нами.

 [Телеграм](#)

 [Подкаст](#)

 [Бесплатные учебники](#)

## Читать дальше

<h1>  
**Отзывы разработчиков, прошедших развитие навыков в HTML Academy**

<h1> + text-wrap: balance;  
**Отзывы разработчиков, прошедших развитие навыков в HTML Academy**

## Новое в 2023

В 2023 в CSS появилось «уравновешивает» текс



**Канал для будущих мидлов**

ачением balance. Оно уравнивались внутри

блока.

```
h1 {  
    text-wgap: balance;  
}
```

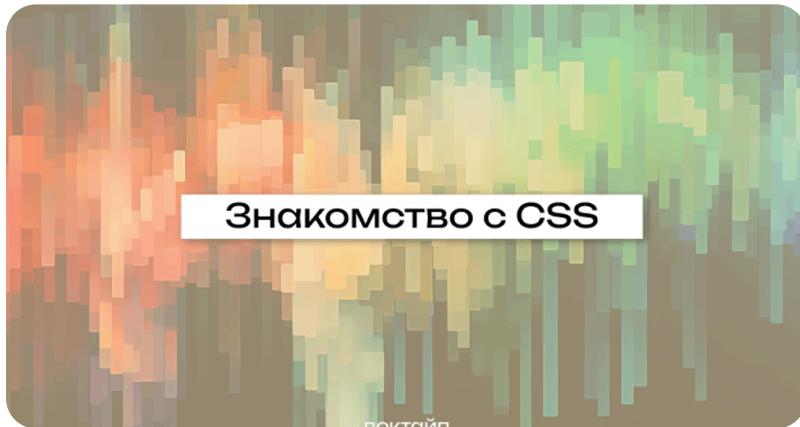
**Ограничение** – текст не длиннее 6 строк, иначе браузеру придётся непросто, и лучше не применять это свойство к `body`.

Вот пример заголовка `<h1>` с `text-wgap: balance` и без него.

На момент написания заметки свойство [поддерживается](#) во всех больших браузерах, кроме Safari, а на мобильных – только в Chrome, но то ли ещё будет.

CSS

13 ноября 2023



## Знакомство с CSS

После того как мы разобрались [с базовой структурой веб-страницы](#) с помощью HTML, пришло время привнести в неё стиль и красоту. В этом нам поможет CSS, что означает *Cascading Style Sheets*, или «каскадные таблицы стилей».

CSS используется для оформления HTML-страниц. Если HTML – это скелет сайта, то CSS – его одежда. С помощью CSS мы можем задавать цвета, шрифты, отступы, добавлять анимации и многое другое.

Читать дальше

CSS



Канал для будущих мидлов

1 ноября 2023

## Я ссылка

оригинал

`transform: scale(1.2);`

## Я ссылка

### Увеличение ссылки при наведении

**Задача:** плавно увеличить ссылку при наведении.

**Решение:**

```
a {  
    display: inline-block;  
    transition: transform 0.3s ease;  
}  
a:hover {  
    transform: scale(1.2);  
}
```

Первые два свойства просто немного меняют вид ссылки. Свойство `color: magenta;` меняет цвет текста в тегах `<a>` на темно-красный, а свойство `text-decoration: none;` убирает подчеркивание.

Но наша задача — плавно увеличить размер ссылки, а не просто её перекрасить. Поэтому используем свойство `transform: scale(1.2)`, которое срабатывает при наведении курсора и увеличивает размер ссылки в 1.2 раза по сравнению с её начальным размером.

**Демо:**



**Канал для будущих мидлов**

[HTML](#)[CSS](#)

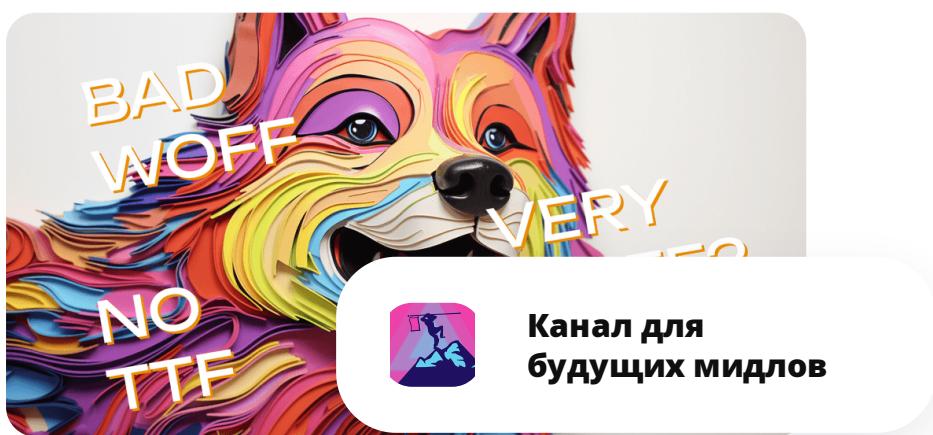
```
a {  
    color: maroon;  
    text-decoration: none;  
    display: inline-block;  
    transition: transform 0.3s ease;  
}  
a:hover {  
    transform: scale(1.2);  
}
```

Увеличение ссылки при наведении

Рекомендуем тренировки: [<a>](#) [color](#) [text-decoration](#)

Узнайте больше о теге [<a>](#)

Подробнее о свойствах [transform](#) [transition](#) [display](#) [color](#) [:hover](#) [text-decoration](#)



# WOFF больше не нужен

Я купил и скачал шрифты для недавнего проекта, распаковал папку, где были только WOFF2-файлы, и сначала не поверил, что такое бывает.

Потом мне стало интересно: они что, забыли WOFF? А он вообще ещё нужен? Ну, всё-таки, веб — это место, где постоянно всё меняется и улучшается, поэтому я пошёл и спросил людей в Mastodon. Ответ был единодушным: нужен только WOFF2!

Я хорошо помню пост от Зака в конце 2016, после которого я отказался от исчерпывающего синтаксиса @font-face, включавшего, вдобавок, TTF, EOT и SVG-шрифты, и перешёл только на WOFF2 и WOFF.

Похоже, с тех пор мир веб-шрифтов изменился ещё разок, и вот актуальная версия @font-face:

```
@font-face {
    font-family: "Open Sans";
    src: url("opensans.woff2") format('woff2');

}
```

Остался всего один формат. Просто, скажите?

Как писал Зак, «так как в вебе, когда шрифт не найден, всё равно подгружаются системные шрифты, мы можем идти в ногу со временем». Итак, какие браузеры отправятся в тёмные века системных шрифтов с этим синтаксисом?

- IE 11, 10, 9, 8, 7, …
- Chrome 4–35
- Edge 12 и 13
- Safari 3–9.1
- Firefox 2–38
- Opera 22 и ниже
- Android 4.4.4 KitKat и ниже (а это <0.5% устройств на Android)
- Safari на iOS 3.2–<sup>r</sup>

Caniuse.com показывает с поддержкой WOFF2. А



**Канал для будущих мидлов**

браузер *Native – прим.*

перев.) заметно, что массовый переход на WOFF2 случился в 2015 и 2016. К концу 2016 во всех последних версиях больших браузеров появилась поддержка WOFF2.3

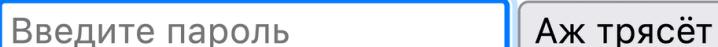
А спустя 7 лет поддержка расширилась настолько, что можно уже не добавлять в проект WOFF-файлы — ну, кроме случая, когда вы точно знаете, что много ваших пользователей используют старые устройства и браузеры.

С другой стороны, нет смысла и удалять WOFF из старых проектов. Если вы подключали WOFF2 *раньше* WOFF внутри @font-face — и порядок здесь важен — то браузер просто скачает и подключит WOFF2-версию.

И если однажды вы, как и я, обнаружите себя перед папкой, полной файлов WOFF2, знайте, что WOFF — уже всё.

CSS

23 сентября 2023



## Трясём пароль с помощью CSS

Знаете момент, когда всё на сайте уже сделано, и хочется добавить какую-нибудь маленькую незаметную фишку? Мы тоже знаем, поэтому давайте просто потрясём поле пароля, когда пользователь ввёл неверный пароль. Как на Маке.

Вот что получится в итоге:



**Канал для будущих мидлов**

HTML

CSS

```
.input-container {
  position: relative;
  width: 300px;
}

@keyframes shake {
  10%, 90% {
    transform: translateX(-0.5px);
  }
  20%, 80% {
    transform: translateX(1px);
  }
  30%, 50%, 70% {
    transform: translateX(-2px);
  }
  40%, 60% {

```

Введите пароль

Submit

## Заголовок

Абзац текста. Lorem ipsum dolor sit amet, consectetur adipiscing elit.

- Элемент списка 1
- Элемент списка 2
- Элемент списка 3

Блок текста в рамке.

## Как сделать темную тему на сайте

Без лишних слов создаётся темная тема с использованием HTML



**Канал для будущих мидлов**

и темной темы при файла —

index.html, styles.css и script.js.

## HTML

Основная разметка страницы — заголовок, абзац текста, список и текст в рамке.

```
<!DOCTYPE html>
<html lang="ru">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Переключатель темы</title>
    <link rel="stylesheet" href="styles.css">
</head>
<body>
    <button id="themeToggle">Переключить тему</button>
    <h1>Заголовок</h1>
    <p>Абзац текста. Lorem ipsum dolor sit amet, consectetur adipiscing elit.</p>
    <ul>
        <li>Элемент списка 1</li>
        <li>Элемент списка 2</li>
        <li>Элемент списка 3</li>
    </ul>
    <div class="boxed-text">
        Блок текста в рамке.
    </div>
    <script src="script.js"></script>
</body>
</html>
```

## CSS (styles.css):

Здесь задаём цвета для светлой и тёмной темы, а ещё минимальную стилизацию текста и блока с рамкой.



**Канал для будущих мидлов**

```

body {
    font-family: Arial, sans-serif;
    transition: background-color 0.3s ease;
}

body.light-theme {
    background-color: #ffffff;
    color: #000000;
}

body.dark-theme {
    background-color: #121212;
    color: #ffffff;
}

.boxed-text {
    border: 2px solid currentColor;
    padding: 15px;
    margin-top: 20px;
}

```

## JavaScript (script.js)

Этот код нужен, чтобы переключать тему при нажатии на кнопку:

```

document.getElementById('themeToggle').addEventListener('click', function() {
    const currentTheme = document.body.className;
    if (currentTheme === 'light-theme') {
        document.body.className = 'dark-theme';
    } else {
        document.body.className = 'light-theme';
    }
});

```

При загрузке страницы по умолчанию будет установлена светлая тема. При нажатии на кнопку «Переключить тему» будет происходить переключение между светлой и темной темой.



**Канал для будущих мидлов**

HTML

CSS

```
body {  
    font-family: Arial, sans-serif;  
    transition: background-color 0.3s ease;  
}  
body.light-theme {  
    background-color: #f2f2f2;  
    color: #000000;  
}  
body.dark-theme {  
    background-color: #2f2f2f;  
    color: #ffffff;  
}  
.boxed-text {  
    border: 2px solid currentColor;  
    padding: 15px;  
}
```

Переключить тему

## Заголовок

Абзац текста. Lorem ipsum dolor sit amet, consectetur adipiscing elit.

- Элемент списка 1
- Элемент списка 2
- Элемент списка 3

Блок текста в рамке.

по центру

## 4 способа центрирования

Центрирование элементов в CSS  
распространенных задачах



Канал для будущих мидлов

CSS

олее  
ьте с макетами.

И хотя центрирование текста по горизонтали довольно простое (`text-align: center;` и делов-то), вертикальное центрирование может быть немного сложнее. Давайте рассмотрим несколько методов.

## Метод 1: Flexbox

Flexbox — это один из самых простых и эффективных способов центрирования.

Заворачиваем текст в `<div>` с классом `center-both`:

```
<div class="center-both">  
  <p>Центрированный текст</p>  
</div>
```

И флексим:

```
.center-both {  
  display: flex;  
  justify-content: center;  
  align-items: center;  
}
```

Получается так:

HTML    CSS

```
.center-both {  
  display: flex;  
  justify-content: center;  
  align-items: center;  
  border: solid orange;  
  height: 150px;  
}
```



Канал для  
будущих мидлов

Центрированный текст

Рекомендуем тренировки: [<div>](#) [<p>](#)

Узнайте больше о тегах [<div>](#) [<p>](#)

Подробнее о свойствах [flex](#) [display](#) [height](#)

## Метод 2: CSS Grid

HTML такой же, как в предыдущем примере. В CSS включаем гриды и используем свойство `place-items` со значением `center`:

```
.center-both {  
  display: grid;  
  place-items: center;  
}
```

Результат:



Канал для  
будущих мидлов

HTML

CSS

```
.center-both {  
    display: grid;  
    place-items: center;  
    border: solid orange;  
    height: 150px;  
}
```

Центрированный текст

Рекомендуем тренировки: [<div>](#) [<p>](#)

Узнайте больше о тегах [<div>](#) [<p>](#)

Подробнее о свойствах [grid](#) [display](#) [height](#)

Работает.

## Метод 3: позиционирование и Transform

Этот метод немного старомодный и работает не идеально. Здесь у `div` устанавливается `relative` позиция. А `<p>` внутри дива мы сдвигаем с помощью абсолютного позиционирования. Не слишком элегантно:



Канал для  
будущих мидлов

```
.center-both {  
    position: relative;  
}  
  
.center-both p {  
    position: absolute;  
    top: 50%;  
    left: 50%;  
    transform: translate(-50%, -50%);  
}
```

HTML остается таким же. Вот что получается:

HTML	CSS
------	-----

```
.center-both {  
    position: relative;  
    border: solid orange;  
    height: 150px;  
}  
.center-both p {  
    position: absolute;  
    top: 50%;  
    left: 50%;  
    transform: translate(-50%, -50%);  
}
```

Центрированный текст



**Канал для  
будущих мидлов**

Рекомендуем тренировки: [<div>](#) [<p>](#)

Узнайте больше о тегах [<div>](#) [<p>](#)

Подробнее о свойствах [transform](#) [position](#) [height](#)

## Плохой метод: использование line-height

Если у вас однострочный текст, вы можете установить `line-height`, равный высоте родительского элемента.

```
.center-both {  
    line-height: 200px; /* Пример высоты */  
    text-align: center;  
}
```

Этот метод не подойдет для многострочного текста. Да и вообще мы очень не рекомендуем так делать, это прям совсем для любителей острых ощущений. Потому что вот:

HTML CSS

```
.center-both {  
    line-height: 200px; /* Пример высоты */  
    text-align: center;  
    border: solid orange;  
    height: 150px;  
}
```



Канал для будущих мидлов



## Как скруглить рамку. CSS-свойство border-radius

CSS-свойство `border-radius` помогает скруглить углы элемента. Оно особенно полезно для стилизации кнопок, форм, карточек товаров и других элементов сайта.

Читать дальше

CSS

28 июля 2023



## CSS-свойство contain

Представьте, что у вас текст, изображения и



**Канал для будущих мидлов**

ся разные элементы: орит браузеру, как

именно элементы должны взаимодействовать. Например, они могут быть ограничены, влиять на расположение друг друга или менять свои размеры.

Также свойство помогает повысить производительность страницы. Например, браузер понимает, когда при изменении свойств элемента нужно перерисовать страницу, а когда нет.

 CSS-свойство `contain` определяет, как элемент должен взаимодействовать с другими элементами внутри контейнера.

## Синтаксис

```
.container {  
  contain: strict;  
}
```

Читать дальше

CSS

14 июля 2023



## Как задать позицию и размер элемента. CSS-свойство `inset`

CSS-свойство `inset` задаёт позицию и размер элемента на странице. Это комбинация четырёх отступов `top`, `bottom`, `right` и `left`, которые определяют отступы от края элемента.



**Канал для будущих мидлов**

## Синтаксис

```
.element {  
  inset: 10px 20px 30px 40px;  
}
```

Читать дальше

CSS

13 июля 2023



**Канал для  
будущих мидлов**

## Практикум

Курсы для новичков

Подписка

## Профессии

Фронтенд-разработчик

JavaScript-разработчик (React)

JavaScript-разработчик (Vue.js)

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

## Услуги

Работа наставником

Для вузов и колледжей

Для учителей

## Журнал

Справочник по HTML

Учебник по Git

Учебник по PHP

## Информация

Об Академии

О центре карьеры

## Остальное

Написать нам

Мероприятия

Форум

Акции

Соглашение

Конфиденциальность

Сведения об образовательной организации

Лицензия № Л035-01271-78/00176657



© 000 «Интерактивные обучающие технологии», 2013–2023



**Канал для будущих мидлов**