

Вёрстка многослойных элементов интерфейса

Выполнен на 51%

Введение

Углублённая теория

Методика вёрстки многослойных элементов

Вёрстка кнопки «Наверх»

Вёрстка «липкой» горизонтальной навигации

Вёрстка «липкого» sidebar

Вёрстка окна с соглашением об использовании Cookie

Вёрстка пользовательской подсказки

Вёрстка компонента Карусель/Слайдер

Вёрстка компонента Выпадающий список

Вёрстка компонента Чат-бот

Вёрстка сложного меню

Вёрстка компонента для расширенного поиска

Вёрстка компонента Tinder card

Вёрстка модального окна

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

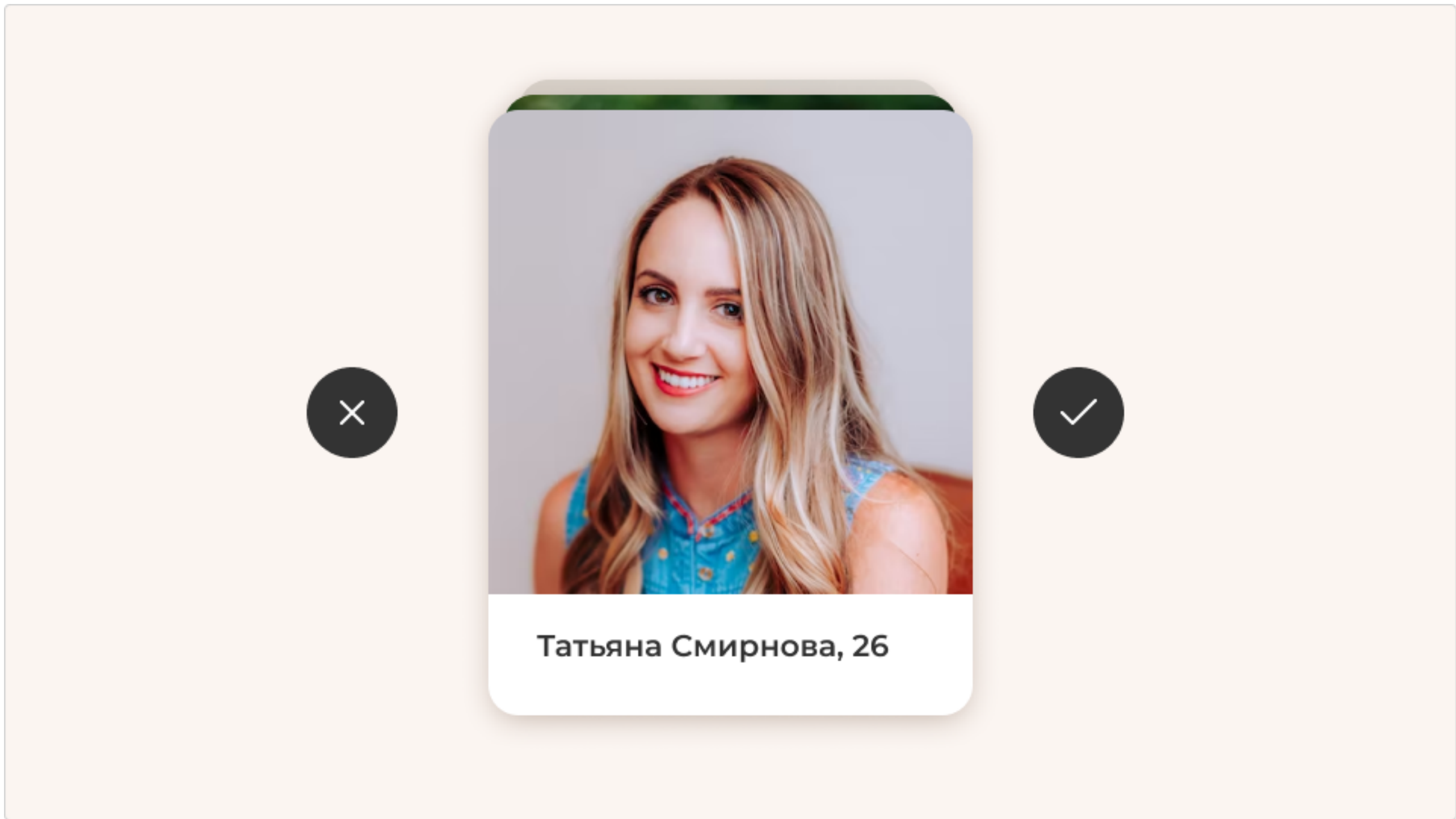
Кейс 9, сложный уровень

Кейс 10, сложный уровень

Главная / Моё обучение / Вёрстка многослойных элементов интерфейса / Методика вёрстки многослойных элементов

Вёрстка компонента Tinder card

Следующий многослойный элемент — это стандартный карточный компонент, который получил своё название из-за всем известного приложения для знакомств *Tinder*. Полное его название — карточный паттерн *Tinder*. Карточки сложены в стопку, причём чем ниже в стопке карточка, тем меньше её размер. Иногда карточки смещены по вертикали, чтобы возникало ощущение трёхмерности. Вы можете поставить отметку *Нравится* или *Не нравится*, смахнув карточку вправо или влево. Порядок карточек в стопке может меняться в зависимости от ваших предпочтений. Эта идея отображения контента сейчас используется в множестве приложений.



Компонент Tinder card

Разберём визуальную составляющую списка карточек и возможную реализацию разметки и стилей.

В разметке набор карточек — это список. Отдельная карточка тянет на роль раздела — заворачиваем её в тег `<section>`.

```
<ul class="catalog-list">
  <li class="catalog-item">
    <section class="card">
      <h3 class="card-name">Татьяна Смирнова, 26</h3>
      
    </section>
  </li>
  <li class="catalog-item">
    <section class="card">
      <h3 class="card-name">Иван Петров, 42</h3>
      
    </section>
  </li>
  <li class="catalog-item">
    <section class="card">
      <h3 class="card-name">Нина Огурцова, 31</h3>
      
    </section>
  </li>
</ul>
```

Размеры у карточек фиксированные, и многослойность можно добавить как с помощью позиционирования, так и с помощью *CSS Grid Layout*.

Для начала опишем, как могут выглядеть примерные стили, если используется позиционирование.

```
.catalog-list {
  list-style: none;
  padding: 0;
  margin: 0;

  position: relative; /* Точка отсчёта для координат карточек */
  width: 320px;      /* Размеры стопки с карточками */
  height: 400px;
}

.catalog-item {
  position: absolute;
  top: 0;
  left: 0;
  width: 320px;
}
```

Для решения с *CSS Grid Layout* стили будут выглядеть примерно так:

```
.catalog-list {
  list-style: none;
  padding: 0;
  margin: 0;

  display: grid;
  justify-content: center; /* По горизонтали выравниваем по центру грид-контейнера */
  align-items: start; /* По вертикали выравниваем вровень с началом грид-контейнера */
  min-height: 400px; /* Зафиксируем высоту контейнера при нулевом количестве карточек */
}

.catalog-item {
  grid-area: 1 / 2 / 1 / 2; /* Все карточки расположены в одной и той же ячейке */
  width: 320px;
}
```

Порядок карточек в стопке определяется их порядком в разметке, последняя карточка в списке будет сверху.

Добавим трёхмерности. В нашем примере карточки отличаются по размерам. Чем ниже карточка в стопке, тем она меньше. Ещё каждая последующая карточка ниже предыдущей.

```
.catalog-item:nth-last-child(3) {
  transform: scaleX(0.875) translateY(-20px);
}

.catalog-item:nth-last-child(2) {
  transform: scaleX(0.9375) translateY(-10px);
}
```

Можно добавить анимацию и после того, как верхнюю карточку смахнут влево или вправо, следующая за ней будет плавно увеличиваться в размерах:

```
.catalog-item {
  ...
  will-change: transform;
  transition: transform 0.3s ease-in-out;
}
```

Замечание

В коде мы использовали CSS-свойство `will-change`, чтобы подсказать браузеру, какое свойство изменится во время анимации. Подробное описание, как работает `will-change` читайте в [спецификации](#).

Вёрстка компонента для расширенного поиска

Вёрстка модального окна

Практикум

Тренажёры

Подписка

Для команд и компаний

Учебник по PHP

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