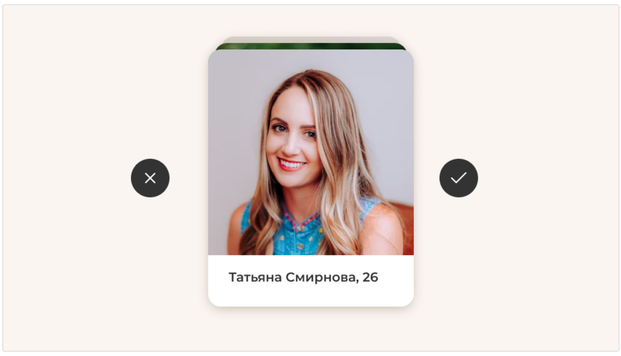
Вёрстка компонента Tinder card

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>
      <img class="card-img" src="img/photo-695.jpg" width="320" height="320" alt="">
    </section>
  </li>
  <li class="catalog-item">
    <section class="card">
      <h3 class="card-name">Иван Петров, 42</h3>
      <img class="card-img" src="img/photo-694.jpg" width="320" height="320" alt="">
    </section>
  </li>
  <li class="catalog-item">
    <section class="card">
      <h3 class="card-name">Нина Огурцова, 31</h3>
      <img class="card-img" src="img/photo-693.jpg" width="320" height="240" alt="">
    </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