

Доклад: "Переходы и анимации в CSS"

Исполнитель 1: Введение и Переходы (Transition)

Что такое переходы?

Переходы в CSS — это способ сделать изменения стилей плавными, а не резкими. Представьте кнопку: вы наводите на неё курсор, и она меняет цвет с серого на синий. Без переходов цвет сменится мгновенно, а с переходами — это будет мягкое и приятное изменение. За это отвечает свойство `transition`.

Основы свойства `transition`

Свойство `transition` говорит браузеру: "Когда стиль элемента меняется, делай это плавно". Оно состоит из четырёх частей:

- `transition-property` — что именно будет меняться. Например:
 - `background-color` — цвет фона.
 - `opacity` — прозрачность, от 0 (невидимый) до 1 (видимый).
 - `transform` — положение или размер (например, сдвиг или увеличение).
- `transition-duration` — сколько времени займёт изменение. Например, `0.5s` — полсекунды.
- `transition-timing-function` — как будет происходить изменение. Например, `ease-in-out` — медленно в начале, быстрее посередине, медленно в конце.
- `transition-delay` — задержка перед началом, например, `0.2s`.

Все значения `transition-property`

Вы можете анимировать почти любой стиль с промежуточными значениями:

- `background-color`, `color` — цвета.
- `opacity` — прозрачность.
- `width`, `height`, `margin`, `padding` — размеры и отступы.
- `transform` — движение, увеличение, поворот.
- `border-width`, `font-size` — толщина границы, размер текста.
- `all` — всё, что меняется; `none` — ничего.

Но нельзя анимировать `display` или `position`.

Все значения `transition-timing-function`

Это "характер" движения:

- `linear` — равномерно.

- `ease` — мягкий старт и финиш (по умолчанию).
- `ease-in` — медленный старт.
- `ease-out` — мед10 — быстрый конец.
- `ease-in-out` — плавно в начале и конце.
- `step-start`, `step-end` — резкие скачки.
- `steps(n)` — шаги (например, `steps(4)` — 4 рывка).
- `cubic-bezier(x1, y1, x2, y2)` — кастомная кривая.

Полное и сокращённое написание

- **Полное:**

```
transition-property: opacity;  
transition-duration: 0.5s;  
transition-timing-function: ease-in-out;  
transition-delay: 0.2s;
```

Прозрачность меняется за полсекунды с задержкой.

- **Сокращённое:**

```
transition: opacity 0.5s ease-in-out 0.2s;
```

Порядок: что, сколько, как, когда.

Примеры:

- Кнопка:

```
transition: background-color 0.5s ease-in-out, opacity 0.5s linear;  
:hover { background-color: blue; opacity: 0.7; }
```

- Увеличение:

```
transition: transform 0.3s ease;  
:hover { transform: scale(1.1); }
```

- `transform` — двигает или меняет элемент.
 - `scale(1.1)` — увеличивает в 1,1 раза.
-

а еще существует JavaScript событие срабатывающее после окончания `Transition` оно называется `Transitioned`

Исполнитель 2: Анимации (@keyframes), Свойство Animation, Loader, Сложные анимации и Заключение

Введение в анимации

Добрый день, уважаемые коллеги!

Я продолжу доклад и расскажу про анимации — более сложный инструмент CSS. Мы разберём свойство `animation`, загрузочный индикатор (loader), примеры анимаций с JavaScript и подведём итоги.

Что такое анимации?

Анимации создают движения, которые работают сами по себе. Вы задаёте этапы с помощью `@keyframes` — это правило, а не функция, — а запускаете их свойством `animation`.

Как работает @keyframes ?

Пример:

```
@keyframes slide {
  0% { transform: translateX(0); opacity: 1; }
  100% { transform: translateX(100px); opacity: 0; }
}
```

- `translateX(100px)` — сдвиг вправо на 100 пикселей.
- `opacity` — от 1 (видимый) до 0 (невидимый).

Свойство animation

`animation` запускает анимацию из `@keyframes`. Оно объединяет до восьми параметров:

- `animation-name` — имя анимации (например, `slide`).
- `animation-duration` — длительность (например, `2s`).
- `animation-timing-function` — скорость (например, `ease-in-out`).
- `animation-delay` — задержка (например, `0.5s`).
- `animation-iteration-count` — повтор (например, `infinite` или `3`).
- `animation-direction` — направление:
 - `normal` — от 0% к 100%.
 - `reverse` — от 100% к 0%.
 - `alternate` — туда и обратно.
- `animation-fill-mode` — до и после:

- `forwards` — остаётся в конце.
- `backwards` — начинается с начала.
- **animation-play-state** — `running` (идёт) или `paused` (на паузе).

Пример:

```
animation: slide 2s ease-in-out 0.5s infinite;
```

Элемент едет вправо и исчезает с задержкой, повторяясь бесконечно.

Пример: Пульсация

```
animation: pulse 1s infinite alternate;
@keyframes pulse {
  0% { transform: scale(1); opacity: 1; }
  100% { transform: scale(1.2); opacity: 0.7; }
}
```

Элемент растёт и становится полупрозрачным, затем возвращается.

Анимация Loader

Loader показывает загрузку. Пример круга:

```
.loader {
  width: 50px;
  height: 50px;
  border: 5px solid #ccc;
  border-top: 5px solid #3498db;
  border-radius: 50%;
  animation: spin 1s infinite linear;
}
@keyframes spin {
  0% { transform: rotate(0deg); }
  100% { transform: rotate(360deg); }
}
```

Сложные анимации с JavaScript

CSS хорош для простого, а для сложного нужен JS:

- **Клик:**

```
.box { transition: transform 1s ease; }
.box.move { transform: translateX(200px); }
```

```
button.addEventListener('click', () => box.classList.toggle('move'));
```

- **Последовательность:**

```
setTimeout(() => { circle.style.transform = 'translateX(100px)'; }, 0);  
setTimeout(() => { circle.style.opacity = '0'; }, 500);
```

- **Прокрутка:**

```
window.addEventListener('scroll', () => {  
  if (boxPosition < windowHeight) box.classList.add('visible');  
});
```

Сравнение

- Переходы: Простые изменения при действиях.
- Анимации: Сложные движения, работают сами.

Заключение

Мы разобрали переходы с `transition`, где `ease-in-out` делает их плавными, и анимации с `@keyframes` и `animation`, где `translateX`, `scale`, `opacity`, `infinite` создают эффекты. Вы узнали про `loader` и как JS помогает с сложными задачами. Попробуйте сами: сделайте кнопку с переходом или `loader` с вращением. Спасибо и успехов в CSS!

Примечания

- `animation` добавлено во вторую часть (Исполнитель 2) после `@keyframes`, с объяснением всех параметров и примерами.
- Доклад остаётся сбалансированным: Исполнитель 1 — переходы, Исполнитель 2 — анимации, включая `animation`, `loader` и JS.
- Если нужно что-то сократить или выделить для слайдов — дай знать!