CSS Animations

Working & Practical Applications

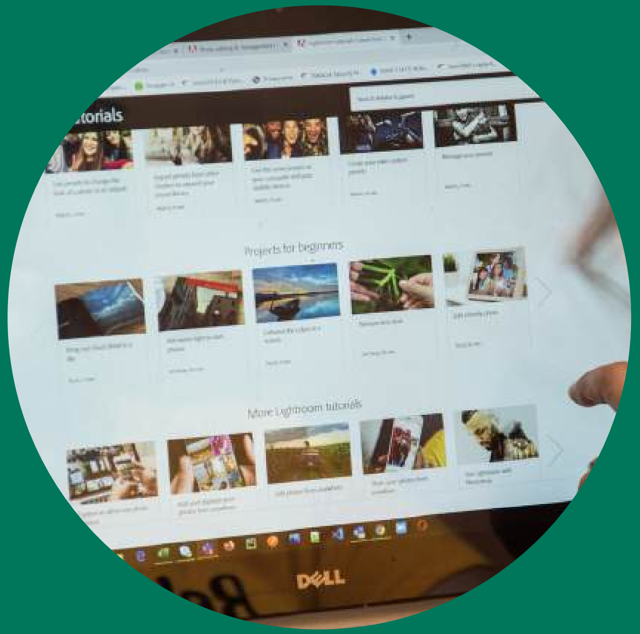
POINTS TO COVER :-

What is CSS Animation?
Main concepts & Fundamentals

How the CSS works?
Browser rendering & Performance

Practical Applications
Real world use and cases

Resources &Tools
Codes & Live examples



WHAT IS CSS ANIMATION?

- CSS animation is that which allows animation of HTML elements without using JavaScript
- An animation lets an element gradually change from one style to another.
- In CSS animation, you must specify keyframes for the animation.
- Keyframes are what styles the element will have at certain times.

@keyframes

Define animation stages with property changes at specific points

animation property

Apply animations to elements with timing, duration, and behavior

Keyframe

```
@keyframes fadeIn {
  0%   { opacity: 0; }
  100% { opacity: 1; }
}
```

```
@keyframes myAnimation {
  0%   { background-color: red;    }
25%  { background-color: yellow; }
50%  { background-color: blue;   }
  100% { background-color: green;  }
}
```

Element

```
.box {
  animation-name:           fadeIn;
  animation-duration:        2s;
  animation-timing-function: ease-in-out;
  animation-delay:          0.5s;
  animation-iteration-count: 1;
}
```

Could also be written as:

```
animation: fadeIn 2s ease-in-out 0.5s 1;
```

```
div {
 width: 100px;
 height: 100px;
 background-color: red;
 animation-name: myAnimation;
 animation-duration: 4s;
 animation-fill-mode: forwards;
}
```

Keyframe

```
@keyframes myAnimation {
  0%  { background-color: red;
transform: translate(0px, 0px); }  25% {
background-color: yellow;
transform: translate(200px, 0px); }
50% { background-color: blue;
transform: translate(200px, 200px); }
75% { background-color: green;
transform: translate(0px, 200px); }
100%{ background-color: red;
transform: translate(0px, 0px); }
}
```

Element

```
div {

 width: 100px;  height: 100px;
 background-color: red;
 animation-name: myAnimation;
 animation-duration: 4s;
animation-timing-function:linear;
animation-iteration-count: infinite;
}
```

KEY ANIMATION PROPERTIES

animation-duration	How long the animation takes	`2s, 500ms`
animation-timing-function	Speed curve of the animation	`ease, linear, cubic-bezier()`
animation-iteration-count	Number of times to repeat	`1, 3, infinite`

HOW BROWSERS RENDER ANIMATION?

- CSS animations run on the browser's rendering engine, potentially using **GPU acceleration for smooth 60fps performance**.

1

Parse CSS

Browser reads @keyframes rules

2

Calculate Values

Interpolates property values between keyframes

3

Composite Layers

GPU-accelerated properties create separate layers

4

Paint & Display

Browser repaints at ~60fps for smooth animation

BEST PRACTICES FOR GOOD PERFORMANCE

GPU-Accelerated Properties

- transform
- opacity
- filter

⚠ Avoid Animating

- width / height
- top / left / margin
- box-shadow (expensive)

- Limit simultaneous animations on complex pages
- Test on mobile devices for performance bottlenecks

PRACTICAL APPLICATIONS UI/UX

Loading States

Spinners, progress bars, skeleton screens

Micro-interactions

Button hover effects, form validation feedback

Page Transitions

Smooth navigation between views

Attention Grabbers

Pulsing notifications, highlighting updates

ADVANCED APPLICATIONS

Responsive Animations

- Adapt to screen sizes
- prefers-reduced-motion media query
- Touch-friendly interactions

Storytelling & Scroll Effects

- Parallax scrolling
- Progressive content reveal
- Interactive narratives

Data Visualization

- Animated charts and graphs
- Transitions between states
- Real-time updates

ANIMATIONS vs TRANSITIONS

CSS Animations

- Define multiple keyframes
- Auto-play on page load
- Loop infinitely or specific count
- Complex timing control
- Better for storytelling

CSS Transitions

- Two states: start and end
- Triggered by state change
- Simpler syntax
- One-time effect
- Better for interactions

COMMON ANIMATION PATTERNS

Fade In/Out

Page elements, modals, tooltips

Slide In/Out

Sidebars, notifications, drawers

Bounce/Pulse

Buttons, alerts, call-to-actions

Rotate/Spin

Loading indicators, icons

Scale/Zoom

Image galleries, hover effects

Shake/Wobble

Form validation errors

LIVE DEMONSTRATION

BROWSER SUPPORT & COMPATIBILITY

✓ CSS animations are widely supported across all modern browsers (Chrome, Firefox, Safari, Edge) with excellent backward compatibility.

Use vendor prefixes for older browsers

`-webkit-animation, -moz-animation`

Provide fallbacks for critical content

`noscript tags, static alternatives`

Test with prefers-reduced-motion

`@media (prefers-reduced-motion: reduce)`

RESOURCES & TOOLS

Learning Platforms

- MDN Web Docs - CSS Animations
- CSS-Tricks Animation Guide
- W3Schools CSS Animation Tutorial

Tools & Libraries

- Animate.css - Ready-made animations
- Animista - Animation generator
- Keyframes.app - Visual timeline editor

Testing & Performance

- Chrome DevTools - Performance tab
- Lighthouse - Audit animations
- Can I Use - Browser compatibility

KEY TAKEAWAYS

1. CSS animations provide smooth, performant motion without JavaScript
2. Use GPU-accelerated properties (transform, opacity) for best performance
3. Combine @keyframes with animation properties for precise control
4. Consider accessibility with prefers-reduced-motion queries
5. Animations enhance UX when used purposefully, not decoratively

QUESTIONS?

THANK YOU FOR ATTENTION