web motion design

TLDR;

□□□□□□□□□□□□□□□□□□□□□□□□cool□□□□□□□□□□□□□□□□□□□□□
□□□□□□□□□

- □□□□
- □□□transition□
- □□□animation□

video background

```
<video autoplay muted loop>
<source src="path/video.mp4" type="video/mp4"><!-- better compatibility --!>
<source src="path/video.webm" type="video/webm"><!-- smaller size --!>
</video>
```

transitions

Simple (state A to B), but NOT handle complex motions.

```
transition:
<transition-property>
<transition-duration>
<transition-function>
<transition-delay:0>,
color .4s ease-in-out 1s;
/* transition-property: all; NOT good for performance */
```

□□ transition-property □□□□□□□□ background□□□□□□□□ Get Around□

```
// □□□□
opacity: 0; // to 1
transform: translate3d(<x>, <y>, <z>);
color
```

object-*

image/video □□□ src □□□ layout

```
object-position // □□
object-fit:
fill |
contain |
cover |
none |
scale-down
```

transform

```
transform-box:
content-box |
border-box |
fill-box | svg: act as content-box for CSS layout elements
stroke-box | svg: act as border-box for CSS layout elements
view-box // svg default, The nearest
```

SVG viewport is used as the reference box

transform-origin: // □□□□

transform-style: // □□□□□□□□□□□□ 3D □□□□□□□□□

flat|
preserve

animations

```
animation:
<duration> <easing-function> <delay> <iteration-count> <direction> <fill-mode> <pl

animation-fill-mode: <[none|forwards|backwards|both]>; // □□□□□□/□□□□□□□□□□□□□□
animation-play-state: <[paused|running]>;
animation-direction: <[normal|reverse|alternate?-reverse]>

@keyframes <name> {
<timing> {}
}
```

□ transition □□□□□□□□□□□□□□□□□□□□
□□□□□□□□□□□□□□□
□□□□□□□□□□□□□□□GSAP□□□ JS□□□□□□□□□□□□□□□□□□

sample: Mac Dock□□□□□□ stick □□

```
.list {
transform-style: preserve-3d;
transform: perspective(1000px);
}
.list .item {
transition: .5s;
filter: brightness(0);
}
.list .item:hover {
filter: brightness(1);
transform: translateZ(200px);
}
.list .item:hover + * { // hover □1□□
filter: brightness(0.6);
transform: translateZ(150px) rotateY(40deg);
}
.list .item:hover + * + * { // hover □2□□
filter: brightness(0.4);
transform: translateZ(70px) rotateY(20deg);
}
.list .item:has(+ *:hover) { // hover □1□□
filter: brightness(0.6);
transform: translateZ(150px) rotateY(40deg);
}
.list .item:has(+ * + *:hover) { // hover □2□□
filter: brightness(0.4);
transform: translateZ(70px) rotateY(20deg);
}
```

sample: svg text loader

```
svg path {
animation: animation-stroke 42 ease-in-out 1 forwards;
}
// □□□□□□□□□□ dash□□—— —— □ -> □ —— ——□
@keyframes animate-stroke {
0% {
fill: transparent;
stroke: some-color;
stroke-width: 3;
stroke-dashoffset: 25%; // dash □□□□
stroke-dasharray: 0 32%; // dash and gap
}
```

```
}

50% {
fill: transparent;
stroke: some-color;
stroke-width: 3;
}

80%, 100% {
fill: some-color;
stroke: transparent;
stroke-width: 0;
stroke-dashoffset: -25%; // dash □□□□
stroke-dasharray: 32% 0; // dash and gap
}
}
```

sample: loading dots

```
.dot-container {
display: flex;
gap: .25rem;
}
.dot-container .dot {
width: .8rem;
height: .8rem;
background-color: white;
border-radius: 50%;
animation: 1s ease-in-out infinite scaleUp;
}

.dot-container .dot:nth-child(2) {
animation-delay: .5s;
}
.dot-container .dot:nth-child(3) {
animation-delay: 1s;
}

@keyframes scaleUp {
0%, 80%, 100% {
transform: scale(0);
}

40% {
transform: scale(1);
}
}
```

sample: animate illustrations

```
[id^="star-"] {
animation: 6s ease-in-out infinite alternate pulse;
transform-origin: center;
}
@keyframes pulse {
0%, 100% {
transform: scale(1.2);
opacity: 1;
}
50% {
transform: scale(0);
opacity: 0;
}
}
@keyframes bounce {
// copy from animate.css
}
```

sample: animate avatar

sample: animate avatar

```
.avatar {
background-image: url("avavar-of-multi-frames");
background-repeat: none;
background-size: cover;
animation: 2s steps(<frames-count - 1>) infinite walking;
}

@keyframes walking {
to {
background-position: 100%;
}
}
```

sample: animate logo

```
<script src="path/to/gsap.js"></script>
<script>
gsap.set("#anchor", {
scale: 0,
transformOrigin: "50% 50%"
});
let tl = gsap.timeline({
repeat: -1, // same as infinite
repeatDelay: 1.5,
yoyo: true // same as alternate
});
tl.to("#anchor", {
scale: 1,
rotation: 360,
duration: 1.2,
ease: "elastic.out"
});
tl.fromTo("#left", {
drawSVG: "100% 100%"
},
{
drawSVG: "0 100%", // stroke start, end
duration: 1.2,
ease: "power4.inOut"
});
</script>
```

sample: typewriter effect

```
let mainTl = gsap.timeline({ repeat: -1 });
let tl = gsap.timeline({
repeat: 1,
yoyo: true,
repeatDelay: 6
});

words.foreach(word => {
tl.to("#typewriter", {
text: word, // core code here
duration: 1
});

mainTl.add()
})

// □□
#cursor {
animation: 1s infinite steps(1) blink;
}

@keyframes blink {
```

```
0%, 100% { opacity: 0 }

50% { opacity: 1 }
}
```

advanced effect samples

- □□□□ effect□□□□□ getBoundClientReact()□□□□□□□□□□□□□□□□□□□□□□

```
gsap.to□magneto, {
duration: 1,
x: computedX, // core code here
y: computedY // core code here
}□
```

- □□□□ effect□□□gsap □□ scrollJS□□
- Parallax animation□□□□□□□□□ layer□□ layer □□

prefers-reduced-motion

```
@media (prefers-reduced-motion: reduce) {
*, ::before, ::after {
animation-duration: 0.001s !important;
animation-iteration-repeat: 1 !important;
transition-duration: 0.001s !important;
}
}
```

GSAP[1]

□□ animation □□□□□□□□□[2]
core code:

- □□□□□click / animationstart / animationend
- Loop□window.requestAnimationFrame

```
/*
To experiment with a different CSS property:
1. Update the lines marked with "start" and "end" in the CSS.
2. Update `animatedPropName` in the JS below.
3. Run the experiment!

Note: the layout CSS of this codepen itself can bork some animations. For example,
*/

const animatedPropName = 'height';

/* you probably don't need to change anything below */

const start = document.getElementById('start');
const subject = document.getElementById('subject');
const output = document.getElementById('output');
const propName = document.getElementById('prop-name');

start.addEventListener('click', startExperiment);
subject.addEventListener('animationstart', startAnimation);
subject.addEventListener('animationend', endAnimation);

propName.innerText = animatedPropName;

let isAnimating = false;
let startTime;

function startExperiment() {
// change button
start.disabled = true;
```

```
  start.innerText = 'Animating...';

  // clear previous experiment
  subject.classList.remove('animate');
  output.innerText = '';

  // begin animation
  addSnapshot('start');
  subject.classList.add('animate');
  }

  function startAnimation() {
  isAnimating = true;
  startTime = Date.now();
  window.requestAnimationFrame(update);
  }

  function endAnimation() {
  // wrap things up
  isAnimating = false;
  addSnapshot('end');

  // change button
  start.disabled = false;
  start.innerText = 'Restart Animation';
  }

  function update() {
  if (isAnimating) {
  const elapsedTime = Date.now() - startTime;
  addSnapshot(elapsedTime + 'ms');
  window.requestAnimationFrame(update);
  }
  }

  function addSnapshot(time) {
  const styles = window.getComputedStyle(subject);
  const propValue = styles.getPropertyValue(animatedPropName);
  output.innerText += time.padStart(5) + ': ' + propValue + '\n';

  // scroll output to bottom
  output.scrollTop = output.scrollHeight;
  }
```

todo□□□□□□□□□□ z-index demo□□□□□□□□□□□□□

---

1. https://gsap.com □
2. https://codersblock.com/blog/the-surprising-things-that-css-can-animate/ □

Date: 2024-03-12
Words: 1283
Time to read: 6 mins

Newer Older

2024-03-25
bundle

2024-03-11
[react] server component