**Grundidee**

* Du hast einen Container (.wrapper) mit deinem Text.
* Darüber liegt ein halbtransparentes Overlay (.overlay), das den Text unsichtbar macht.
* In dieses Overlay schneiden wir mit einer **Maske** (radialer Gradient = oval mit weichen Rändern) ein Loch hinein – dieses Loch folgt der Mausposition.

HTML:

<!DOCTYPE html>

<html lang="de">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Taschenlampen-Effekt</title>

<style>

body {

display: flex;

justify-content: center;

align-items: center;

height: 100vh;

background: #222;

color: white;

font-family: Arial, sans-serif;

}

.wrapper {

position: relative;

width: 300px;

height: 600px;

overflow: hidden;

border: 2px solid white;

}

.content {

padding: 20px;

font-size: 1.2rem;

line-height: 1.5;

}

.overlay {

position: absolute;

top: 0;

left: 0;

width: 100%;

height: 100%;

background: black; /\* verdeckt den Text \*/

/\* Maske: wird per JS verändert \*/

-webkit-mask-image: radial-gradient(ellipse 50px 80px at 50% 50%, rgba(0,0,0,1) 70%, rgba(0,0,0,0) 100%);

-webkit-mask-repeat: no-repeat;

-webkit-mask-position: center;

-webkit-mask-composite: destination-out;

mask-image: radial-gradient(ellipse 50px 80px at 50% 50%, rgba(0,0,0,1) 70%, rgba(0,0,0,0) 100%);

mask-repeat: no-repeat;

mask-position: center;

}

</style>

</head>

<body>

<div class="wrapper">

<div class="content">

<p>„Zeit ist das, was man an der Uhr abliest.“ – Einstein</p>

<p>„Planung ersetzt Zufall durch Irrtum.“</p>

<p>„Wer die Zeit verliert, verliert das Leben.“</p>

<p>„Warte nicht auf die perfekte Zeit – fang an.“</p>

<p>„Ein Ziel ohne Plan ist nur ein Wunsch.“</p>

</div>

<div class="overlay"></div>

</div>

<script>

const overlay = document.querySelector('.overlay');

document.querySelector('.wrapper').addEventListener('mousemove', (e) => {

const rect = e.currentTarget.getBoundingClientRect();

const x = e.clientX - rect.left;

const y = e.clientY - rect.top;

overlay.style.maskImage = `radial-gradient(ellipse 50px 80px at ${x}px ${y}px, rgba(0,0,0,1) 70%, rgba(0,0,0,0) 100%)`;

overlay.style.webkitMaskImage = `radial-gradient(ellipse 50px 80px at ${x}px ${y}px, rgba(0,0,0,1) 70%, rgba(0,0,0,0) 100%)`;

});

</script>

</body>

</html>

**Erklärung für dich (Anfängerfreundlich):**

1. **.overlay** ist wie eine schwarze Decke über deinem Text.
2. Wir benutzen mask-image → stellt dir das wie ein „Loch“ in dieser Decke vor.
3. Mit radial-gradient(ellipse ...) machen wir dieses Loch oval mit weichen Rändern.
4. Mit JS aktualisieren wir bei jeder Mausbewegung die Position des Loches (at x y).

So entsteht der Effekt, dass der Nutzer mit der Maus die Sprüche „freilegen“ kann – wie mit einer Taschenlampe 🔦.

**3. Extra Spielerei: Blur-Effekt am Rand**

Falls du es noch weicher und „mystischer“ machen willst:  
einfach den radial-gradient etwas länger auslaufen lassen, z. B.:

CSS:

radial-gradient(ellipse 60px 100px at 50% 50%, rgba(0,0,0,1) 50%, rgba(0,0,0,0) 100%)

Komplette Taschenlampe Effekt:

<!DOCTYPE html>

<html lang="de">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Taschenlampen-Effekt</title>

<style>

body {

display: flex;

justify-content: center;

align-items: center;

height: 100vh;

background: #222;

color: white;

font-family: Arial, sans-serif;

}

.wrapper {

position: relative;

width: 300px;

height: 600px;

overflow: hidden;

border: 2px solid white;

}

.content {

padding: 20px;

font-size: 1.2rem;

line-height: 1.5;

}

.overlay {

position: absolute;

inset: 0;

background: black; /\* verdeckt den Text \*/

/\* Anfangsmaske, wird per JS überschrieben \*/

-webkit-mask-image: radial-gradient(

ellipse 50px 80px at 50% 50%,

rgba(0,0,0,1) 70%,

rgba(0,0,0,0) 100%

);

-webkit-mask-repeat: no-repeat;

mask-image: radial-gradient(

ellipse 50px 80px at 50% 50%,

rgba(0,0,0,1) 70%,

rgba(0,0,0,0) 100%

);

mask-repeat: no-repeat;

}

</style>

</head>

<body>

<div class="wrapper">

<div class="content">

<p>„Zeit ist das, was man an der Uhr abliest.“ – Einstein</p>

<p>„Planung ersetzt Zufall durch Irrtum.“</p>

<p>„Wer die Zeit verliert, verliert das Leben.“</p>

<p>„Warte nicht auf die perfekte Zeit – fang an.“</p>

<p>„Ein Ziel ohne Plan ist nur ein Wunsch.“</p>

</div>

<div class="overlay"></div>

</div>

<script>

const overlay = document.querySelector('.overlay');

const wrapper = document.querySelector('.wrapper');

wrapper.addEventListener('mousemove', (e) => {

const rect = wrapper.getBoundingClientRect();

const x = e.clientX - rect.left;

const y = e.clientY - rect.top;

// Maske dynamisch an Mausposition setzen

overlay.style.maskImage =

`radial-gradient(ellipse 50px 80px at ${x}px ${y}px, rgba(0,0,0,1) 70%, rgba(0,0,0,0) 100%)`;

overlay.style.webkitMaskImage =

`radial-gradient(ellipse 50px 80px at ${x}px ${y}px, rgba(0,0,0,1) 70%, rgba(0,0,0,0) 100%)`;

});

</script>

</body>

</html>

Ergebnis:

* Die Sprüche sind standardmäßig **verdeckt**.
* Wenn du mit der Maus über den Bereich fährst, folgt dir eine **Taschenlampe mit weichen Rändern** und legt die Texte frei.
* Der Effekt ist **immer aktiv**, solange die Maus über dem Div ist.

Taschenlampe mit weichem Übergang

<!DOCTYPE html>

<html lang="de">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Taschenlampen-Effekt</title>

<style>

body {

display: flex;

justify-content: center;

align-items: center;

height: 100vh;

background: #222;

color: white;

font-family: Arial, sans-serif;

}

.wrapper {

position: relative;

width: 300px;

height: 600px;

overflow: hidden;

border: 2px solid white;

}

.content {

padding: 20px;

font-size: 1.2rem;

line-height: 1.5;

}

.overlay {

position: absolute;

inset: 0;

background: black;

-webkit-mask-image: radial-gradient(

ellipse 50px 80px at 50% 50%,

rgba(0,0,0,1) 70%,

rgba(0,0,0,0) 100%

);

-webkit-mask-repeat: no-repeat;

mask-image: radial-gradient(

ellipse 50px 80px at 50% 50%,

rgba(0,0,0,1) 70%,

rgba(0,0,0,0) 100%

);

mask-repeat: no-repeat;

}

</style>

</head>

<body>

<div class="wrapper">

<div class="content">

<p>„Zeit ist das, was man an der Uhr abliest.“ – Einstein</p>

<p>„Planung ersetzt Zufall durch Irrtum.“</p>

<p>„Wer die Zeit verliert, verliert das Leben.“</p>

<p>„Warte nicht auf die perfekte Zeit – fang an.“</p>

<p>„Ein Ziel ohne Plan ist nur ein Wunsch.“</p>

</div>

<div class="overlay"></div>

</div>

<script>

const overlay = document.querySelector('.overlay');

const wrapper = document.querySelector('.wrapper');

let targetX = 150, targetY = 300; // Zielposition (Mitte)

let currentX = targetX, currentY = targetY; // Startposition

wrapper.addEventListener('mousemove', (e) => {

const rect = wrapper.getBoundingClientRect();

targetX = e.clientX - rect.left;

targetY = e.clientY - rect.top;

});

function animate() {

// langsames Nachziehen (0.1 = Geschwindigkeit)

currentX += (targetX - currentX) \* 0.1;

currentY += (targetY - currentY) \* 0.1;

overlay.style.maskImage =

`radial-gradient(ellipse 50px 80px at ${currentX}px ${currentY}px, rgba(0,0,0,1) 70%, rgba(0,0,0,0) 100%)`;

overlay.style.webkitMaskImage =

`radial-gradient(ellipse 50px 80px at ${currentX}px ${currentY}px, rgba(0,0,0,1) 70%, rgba(0,0,0,0) 100%)`;

requestAnimationFrame(animate);

}

animate(); // Startet die Animation

</script>

</body>

</html>

**Erklärung**

* targetX / targetY = Mausposition.
* currentX / currentY = tatsächliche Lampe.
* In animate() wandern wir jedes Mal **10% näher an die Zielposition** → ergibt einen schönen weichen „Lag“-Effekt.
* requestAnimationFrame sorgt dafür, dass es flüssig läuft.

Taschenlampe mit weichem Übergang + sanftem Einblenden

<!DOCTYPE html>

<html lang="de">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Taschenlampen-Effekt</title>

<style>

body {

display: flex;

justify-content: center;

align-items: center;

height: 100vh;

background: #222;

color: white;

font-family: Arial, sans-serif;

}

.wrapper {

position: relative;

width: 300px;

height: 600px;

overflow: hidden;

border: 2px solid white;

}

.content {

padding: 20px;

font-size: 1.2rem;

line-height: 1.5;

}

.overlay {

position: absolute;

inset: 0;

background: black;

/\* Start: unsichtbar \*/

opacity: 0;

transition: opacity 0.6s ease;

-webkit-mask-image: radial-gradient(

ellipse 50px 80px at 50% 50%,

rgba(0,0,0,1) 70%,

rgba(0,0,0,0) 100%

);

-webkit-mask-repeat: no-repeat;

mask-image: radial-gradient(

ellipse 50px 80px at 50% 50%,

rgba(0,0,0,1) 70%,

rgba(0,0,0,0) 100%

);

mask-repeat: no-repeat;

}

/\* Aktiv: Lampe sichtbar \*/

.overlay.active {

opacity: 1;

}

</style>

</head>

<body>

<div class="wrapper">

<div class="content">

<p>„Zeit ist das, was man an der Uhr abliest.“ – Einstein</p>

<p>„Planung ersetzt Zufall durch Irrtum.“</p>

<p>„Wer die Zeit verliert, verliert das Leben.“</p>

<p>„Warte nicht auf die perfekte Zeit – fang an.“</p>

<p>„Ein Ziel ohne Plan ist nur ein Wunsch.“</p>

</div>

<div class="overlay"></div>

</div>

<script>

const overlay = document.querySelector('.overlay');

const wrapper = document.querySelector('.wrapper');

let targetX = 150, targetY = 300;

let currentX = targetX, currentY = targetY;

wrapper.addEventListener('mouseenter', () => {

overlay.classList.add('active'); // einblenden

});

wrapper.addEventListener('mouseleave', () => {

overlay.classList.remove('active'); // ausblenden

});

wrapper.addEventListener('mousemove', (e) => {

const rect = wrapper.getBoundingClientRect();

targetX = e.clientX - rect.left;

targetY = e.clientY - rect.top;

});

function animate() {

currentX += (targetX - currentX) \* 0.1;

currentY += (targetY - currentY) \* 0.1;

overlay.style.maskImage =

`radial-gradient(ellipse 50px 80px at ${currentX}px ${currentY}px, rgba(0,0,0,1) 70%, rgba(0,0,0,0) 100%)`;

overlay.style.webkitMaskImage =

`radial-gradient(ellipse 50px 80px at ${currentX}px ${currentY}px, rgba(0,0,0,1) 70%, rgba(0,0,0,0) 100%)`;

requestAnimationFrame(animate);

}

animate();

</script>

</body>

</html>

**Jetzt passiert:**

* **Maus betritt den Container** → Overlay wird eingeblendet (opacity: 1).
* **Maus verlässt den Container** → Overlay blendet wieder aus (opacity: 0).
* Die Lampe folgt weiterhin weich mit „Lag“.

**getBoundingClientRect()**

Das ist eine eingebaute Methode in JavaScript, die dir die **Größe und Position eines Elements im Browser-Fenster** zurückgibt.  
Wenn du element.getBoundingClientRect() aufrufst, bekommst du ein **Objekt** mit Werten wie:

const rect = wrapper.getBoundingClientRect();

console.log(rect);

Ergebnis sieht etwa so aus (Beispiel):

{

"x": 100,

"y": 200,

"width": 300,

"height": 600,

"top": 200,

"right": 400,

"bottom": 800,

"left": 100

}

Das bedeutet:

* x und y → Position des Elements relativ zum **Viewport** (also sichtbarer Bereich im Browser).
* width und height → Breite und Höhe des Elements.
* top, left, right, bottom → genaue Abstände zum Fenster.

👉 Warum brauchen wir das?  
Weil die Maus-Koordinaten (e.clientX, e.clientY) immer **bezogen auf das ganze Fenster** sind.  
Wir wollen aber die Position **innerhalb des Divs**.  
Darum ziehen wir die rect.left und rect.top ab:

const x = e.clientX - rect.left; // Maus X innerhalb des Div

const y = e.clientY - rect.top; // Maus Y innerhalb des Div

So bekommen wir die Mausposition **im Container selbst**, nicht im ganzen Fenster.

**🔹 requestAnimationFrame()**

Ja 🎉 – das ist eine **eingebaute JavaScript-Funktion**, extra fürs Animieren.  
Sie sagt dem Browser:  
👉 „Führe diese Funktion beim nächsten Frame (Bild) der Animation aus.“

Ein Browser läuft normalerweise mit **60 FPS (Frames pro Sekunde)** → also 60 Bilder/Sekunde.  
Mit requestAnimationFrame kriegst du einen **flüssigen Ablauf** und der Browser passt es automatisch an die Bildwiederholrate an.

Ein Beispiel:

function animate() {

console.log("Ein Frame");

requestAnimationFrame(animate); // immer wieder aufrufen

}

animate();

Das schreibt dir 60x pro Sekunde „Ein Frame“ in die Konsole.

👉 Vorteil gegenüber setInterval oder setTimeout:

* Läuft synchron zum Bildschirm → viel flüssiger.
* Spart Leistung → pausiert automatisch, wenn das Tab im Hintergrund ist.
* Perfekt für Animationen (wie unser Taschenlampen-Effekt).

🔑 In deinem Code benutzen wir requestAnimationFrame(animate), damit die Lampe **immer wieder ein kleines Stück Richtung Mausposition wandert** → dadurch sieht es weich und sanft aus.

**Was ist ein Frame?**

Ein **Frame** ist einfach **ein Einzelbild einer Animation** – so wie ein Standbild in einem Daumenkino.

* Wenn du ein Video siehst, siehst du **viele Einzelbilder nacheinander**, z. B. 30 oder 60 pro Sekunde.
* Genau dasselbe passiert im Browser bei Animationen: **der Bildschirm wird immer wieder neu gezeichnet** → jedes Mal nennt man das einen Frame.

**Warum 60 FPS?**

* Moderne Bildschirme zeigen **ca. 60 Bilder pro Sekunde** (FPS = Frames per Second).
* Das bedeutet: **jede 1/60 Sekunde** wird ein neuer Frame gezeichnet → Bewegungen wirken flüssig.
* Weniger Frames → Bewegung ruckelig.
* Mehr Frames → noch flüssiger, aber Browser oder CPU müssen mehr leisten.

**Analogie**

Stell dir vor, du hältst ein Daumenkino:

* Jedes Blatt ist ein **Frame**.
* Schnell genug durchblättern → Animation flüssig.
* Langsam → Animation ruckelt.