Trong CSS, `@keyframes` là một quy tắc được sử dụng để định nghĩa các bước chuyển đổi (hoặc khung hình) của một \*\*animation\*\*. Nó cho phép bạn tạo ra các hiệu ứng chuyển động mượt mà bằng cách xác định các trạng thái khác nhau của một thuộc tính CSS tại các thời điểm cụ thể trong thời gian chạy của một quá trình hoạt ảnh (animation).

### Cách làm việc của `@keyframes`:

- `@keyframes` định nghĩa cách các thuộc tính CSS thay đổi trong suốt thời gian của một \*\*animation\*\*.

- Cụ thể, bạn cần định nghĩa một cái tên cho animation đó và cài đặt các bước chuyển đổi ở các mốc phần trăm (hoặc sử dụng `from` và `to` để biểu thị điểm bắt đầu và điểm kết thúc).

---

### Cấu trúc cú pháp cơ bản của `@keyframes`:

```css

@keyframes ten-animation {

/\* Điểm bắt đầu \*/

from {

/\* Các thuộc tính CSS tại điểm bắt đầu \*/

}

/\* Điểm kết thúc \*/

to {

/\* Các thuộc tính CSS tại điểm kết thúc \*/

}

/\* Hoặc bạn có thể dùng phần trăm để định nghĩa \*/

0% {

/\* Trạng thái ban đầu \*/

}

50% {

/\* Trạng thái ở giữa \*/

}

100% {

/\* Trạng thái kết thúc \*/

}

}

```

---

### Kết hợp `@keyframes` trong CSS Animation:

Sau khi tạo ra `@keyframes`, bạn sử dụng nó với thuộc tính `animation` (hoặc các thuộc tính liên quan như `animation-name`, `animation-duration`, v.v.) để áp dụng hiệu ứng.

### Ví dụ minh họa:

Giả sử, chúng ta muốn một hộp vuông di chuyển từ trái sang phải.

```html

<!DOCTYPE html>

<html>

<head>

<style>

/\* Tạo animation bằng @keyframes \*/

@keyframes diChuyen {

from {

transform: translateX(0); /\* Ban đầu: ở vị trí gốc \*/

}

to {

transform: translateX(300px); /\* Kết thúc: dịch sang phải 300px \*/

}

}

/\* Áp dụng animation \*/

.box {

width: 100px;

height: 100px;

background-color: red;

animation: diChuyen 2s linear infinite; /\* Áp dụng "diChuyen", thời gian 2 giây, chuyển động mượt, lặp vô hạn \*/

}

</style>

</head>

<body>

<div class="box"></div>

</body>

</html>

```

### Giải thích ví dụ:

1. \*\*@keyframes diChuyen\*\*:

- Chuyển đổi trạng thái từ `translateX(0)` (vị trí gốc) đến `translateX(300px)` (dịch sang phải 300px).

2. \*\*animation\*\*:

- `diChuyen`: Tên của animation áp dụng.

- `2s`: Thời gian chạy của animation là 2 giây.

- `linear`: Tốc độ chuyển động đồng đều.

- `infinite`: Animation lặp lại vô hạn.

3. Kết quả: Một khối vuông đỏ sẽ tự động di chuyển mượt mà từ bên trái sang phải, sau đó lặp lại.

---

### Một số lưu ý khác:

- Bạn có thể sử dụng nhiều khung hình (keyframes) với các giá trị phần trăm như 25%, 50%, 75%, v.v. để tạo ra animation phức tạp hơn.

- Các thuộc tính CSS phổ biến dùng trong animation thường là `transform`, `opacity`, `background-color`, `width`, `height`, v.v.

- Để animation chỉ chạy một lần, bạn có thể bỏ `infinite` và sử dụng `animation-iteration-count: 1`.

Hy vọng câu trả lời này giúp bạn hiểu rõ hơn về `@keyframes` trong CSS! 😊

`@keyframes` là một quy tắc trong CSS được sử dụng để định nghĩa các bước (hoặc khung hình) của một \*\*animation\*\*. Nó cho phép bạn chỉ định cách mà các thuộc tính CSS sẽ thay đổi qua thời gian trong một quá trình hoạt ảnh (animation). Nói cách khác, `@keyframes` giúp thiết kế và kiểm soát quá trình thay đổi trạng thái của các phần tử một cách chi tiết.

---

### Hiểu đơn giản:

- Bạn có thể tưởng tượng một \*\*animation\*\* là một chuỗi hình ảnh (frames), và `@keyframes` xác định nội dung của mỗi khung hình trong chuỗi đó.

- Nó mô tả những gì sẽ xảy ra ở các thời điểm cụ thể trong suốt thời gian thực hiện của animation.

---

### Cách hoạt động:

Khi khai báo một `@keyframes`, bạn định nghĩa:

1. \*\*Thời điểm cụ thể để thay đổi trạng thái\*\*:

- Có thể sử dụng `from` và `to` (để biểu diễn trạng thái bắt đầu và kết thúc).

- Hoặc dùng tỷ lệ phần trăm (%), ví dụ: `0%`, `50%`, `100%`.

2. \*\*Các thuộc tính CSS cần thay đổi ở từng thời điểm\*\*.

Khi một khung hình (frame) được định nghĩa, trình duyệt sẽ tự động "lấp đầy" các trạng thái ở giữa để tạo ra chuyển động mượt mà.

---

### Cấu trúc:

Cú pháp của `@keyframes` như sau:

```css

@keyframes ten-animation {

/\* Xác định tại các khung hình thời gian \*/

0% { /\* thời điểm ban đầu \*/

/\* Các thuộc tính CSS \*/

}

50% { /\* thời điểm giữa \*/

/\* Các thuộc tính CSS \*/

}

100% { /\* thời điểm kết thúc \*/

/\* Các thuộc tính CSS \*/

}

}

```

### Một vài điểm chú ý:

- Bạn có thể chia hoạt ảnh ra làm nhiều khung bằng cách sử dụng các mốc thời gian như 10%, 20%, 50%, v.v.

- `from` thực chất tương đương với `0%` (bắt đầu) và `to` tương đương với `100%` (kết thúc).

---

### Ví dụ thực tế:

#### 1. Chỉ định chuyển động cơ bản (di chuyển mượt từ trái sang phải):

```css

@keyframes moveRight {

0% { transform: translateX(0); }

100% { transform: translateX(300px); }

}

```

#### 2. Hiệu ứng đổi màu phức tạp hơn:

```css

@keyframes changeColor {

0% { background-color: red; }

50% { background-color: yellow; }

100% { background-color: blue; }

}

```

---

### Áp dụng animation với `@keyframes`:

Sau khi định nghĩa `@keyframes`, bạn cần áp dụng nó cho một phần tử HTML thông qua thuộc tính `animation` trong CSS, ví dụ:

```html

<div class="box"></div>

<style>

/\* Định nghĩa khung hình của animation \*/

@keyframes moveAndColor {

0% { transform: translateX(0); background-color: red; }

50% { transform: translateX(150px); background-color: yellow; }

100% { transform: translateX(300px); background-color: blue; }

}

/\* Áp dụng animation cho phần tử \*/

.box {

width: 100px;

height: 100px;

background-color: red;

animation: moveAndColor 3s ease-in-out infinite; /\* Thời gian: 3s, chuyển động mượt, lặp lại vô hạn \*/

}

</style>

```

---

### Kết quả:

- Hộp (div) sẽ di chuyển từ trái sang phải trong \*\*3 giây\*\* và đồng thời thay đổi màu sắc từ đỏ, sang vàng rồi xanh.

- Giá trị `infinite` khiến hiệu ứng lặp lại vô hạn.

---

### Tóm lại:

`@keyframes` là một thành phần mạnh mẽ của CSS, giúp bạn định nghĩa các khung hình trong một hoạt ảnh để tạo ra các hiệu ứng chuyển động và thay đổi trạng thái hấp dẫn.

Trong CSS Flexbox, `flex: 1` là một cách rút gọn để áp dụng các thuộc tính liên quan đến cách phần tử flex được giãn nở hoặc co dần trong container flex.

Cách viết `flex: 1` tương đương với:

```css

flex: 1 1 0%;

```

Trong đó:

1. \*\*`flex-grow: 1`\*\*: Phần tử sẽ giãn nở để chiếm không gian còn lại trong container flex nếu còn trống.

2. \*\*`flex-shrink: 1`\*\*: Phần tử có thể co lại nếu không gian trong container flex bị thiếu.

3. \*\*`flex-basis: 0%`\*\*: Phần tử sẽ có kích thước cơ bản ban đầu là `0`, sau đó kích thước này sẽ được điều chỉnh dựa trên các thuộc tính `flex-grow` và `flex-shrink`.

### Tác dụng của `flex: 1`

- Phần tử sẽ chiếm đều khoảng không gian trống trong container cùng với các phần tử khác cũng có giá trị `flex: 1`.

- Ví dụ: Nếu bạn đặt `flex: 1` cho tất cả các phần tử con trong container flex, thì các phần tử này sẽ chia đều không gian trống trong container, bất kể kích thước nội dung của mỗi phần tử là bao nhiêu.

### Ví dụ minh họa

HTML:

```html

<div class="container">

<div class="item">1</div>

<div class="item">2</div>

<div class="item">3</div>

</div>

```

CSS:

```css

.container {

display: flex;

}

.item {

flex: 1;

border: 1px solid black;

text-align: center;

}

```

Trong ví dụ này:

- Các phần tử có class `item` sẽ chia đều không gian trong container `container`.

- Mỗi phần tử sẽ chiếm 1/3 (một phần bằng nhau) không gian của `container`, bất kể nội dung của chúng có kích thước khác nhau.

Nếu chỉ một phần tử có `flex: 1` và các phần tử khác có giá trị `flex` mặc định, phần tử đó sẽ giãn nở để chiếm tất cả không gian trống trong container.

### Tuy nhiên, bạn cũng có thể sử dụng các giá trị khác cho `flex`, ví dụ `flex: 2` hoặc `flex: 3`, để phân phối không gian theo tỉ lệ (xem thêm về `flex-grow` để hiểu chi tiết).

Trong CSS, `@media` là một quy tắc (rule) được sử dụng để \*\*áp dụng các kiểu CSS (style) khác nhau tùy thuộc vào điều kiện cụ thể\*\*, chẳng hạn như kích thước màn hình, độ phân giải, hướng của thiết bị, và nhiều yếu tố khác.

Nói cách khác, `@media` giúp tạo \*\*thiết kế đáp ứng (responsive design)\*\*, làm cho trang web hiển thị tốt trên nhiều loại thiết bị và kích thước màn hình khác nhau, ví dụ: điện thoại, máy tính bảng, hoặc máy tính để bàn.

---

## Cú pháp cơ bản của `@media`

```css

@media (điều kiện) {

/\* CSS áp dụng cho điều kiện này \*/

selector {

property: value;

}

}

```

---

### Điều kiện phổ biến trong `@media`

Dưới đây là một số điều kiện thường được sử dụng:

1. \*\*`max-width`\*\*: Áp dụng style khi chiều rộng màn hình nhỏ hơn hoặc bằng giá trị chỉ định.

```css

@media (max-width: 768px) {

body {

background-color: lightblue; /\* Áp dụng nếu màn hình nhỏ hơn hoặc bằng 768px \*/

}

}

```

2. \*\*`min-width`\*\*: Áp dụng style khi chiều rộng màn hình lớn hơn hoặc bằng giá trị chỉ định.

```css

@media (min-width: 1024px) {

body {

background-color: pink; /\* Áp dụng nếu màn hình lớn hơn hoặc bằng 1024px \*/

}

}

```

3. \*\*`min-width` và `max-width` kết hợp\*\*: Áp dụng style trong một khoảng chiều rộng màn hình cụ thể.

```css

@media (min-width: 768px) and (max-width: 1024px) {

body {

background-color: yellow; /\* Áp dụng nếu màn hình từ 768px đến 1024px \*/

}

}

```

4. \*\*`orientation`\*\*: Áp dụng style tùy theo hướng màn hình:

- `landscape` (chiều ngang): màn hình rộng hơn chiều cao.

- `portrait` (chiều dọc): màn hình cao hơn chiều rộng.

```css

@media (orientation: portrait) {

body {

background-color: green; /\* Áp dụng nếu màn hình đang xoay theo chiều dọc \*/

}

}

```

5. \*\*`print`\*\*: Áp dụng style khi nội dung được in ra.

```css

@media print {

body {

color: black; /\* Áp dụng khi in nội dung \*/

background: white;

}

}

```

---

### Ví dụ thực tế với Responsive Design

HTML:

```html

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>Responsive Design</title>

<link rel="stylesheet" href="styles.css">

</head>

<body>

<div class="box">Responsive Box</div>

</body>

</html>

```

CSS (`styles.css`):

```css

/\* Style mặc định \*/

.box {

width: 100px;

height: 100px;

background-color: red;

}

/\* Nếu màn hình có chiều rộng nhỏ hơn hoặc bằng 768px \*/

@media (max-width: 768px) {

.box {

background-color: blue;

width: 50px;

height: 50px;

}

}

/\* Nếu màn hình có chiều rộng lớn hơn 1024px \*/

@media (min-width: 1024px) {

.box {

background-color: green;

width: 150px;

height: 150px;

}

}

```

- Khi màn hình nhỏ hơn 768px, box sẽ chuyển sang màu \*\*xanh dương\*\* và thu nhỏ kích thước.

- Khi màn hình lớn hơn 1024px, box sẽ chuyển sang màu \*\*xanh lá\*\* và kích thước lớn hơn.

---

### Lợi ích của `@media`

- \*\*Responsive Design\*\*: Làm cho trang web dễ sử dụng trên nhiều thiết bị.

- \*\*Hiệu suất tốt hơn\*\*: Chỉ áp dụng style cần thiết đối với từng thiết bị.

- \*\*Tương thích với đa dạng thiết bị\*\*: Đáp ứng mọi kích thước màn hình như điện thoại, máy tính bảng, laptop.

`@media` rất quan trọng trong phát triển web hiện đại, đặc biệt khi các trang web cần hoạt động trên nhiều loại thiết bị khác nhau!