**Day 27**





**“Web Development + Security”**

**CSS Transforms:**

**What is transform in CSS?**

The transform property lets you visually change the shape, position, and size of an element without affecting surrounding elements.

**Example:**

*div {*

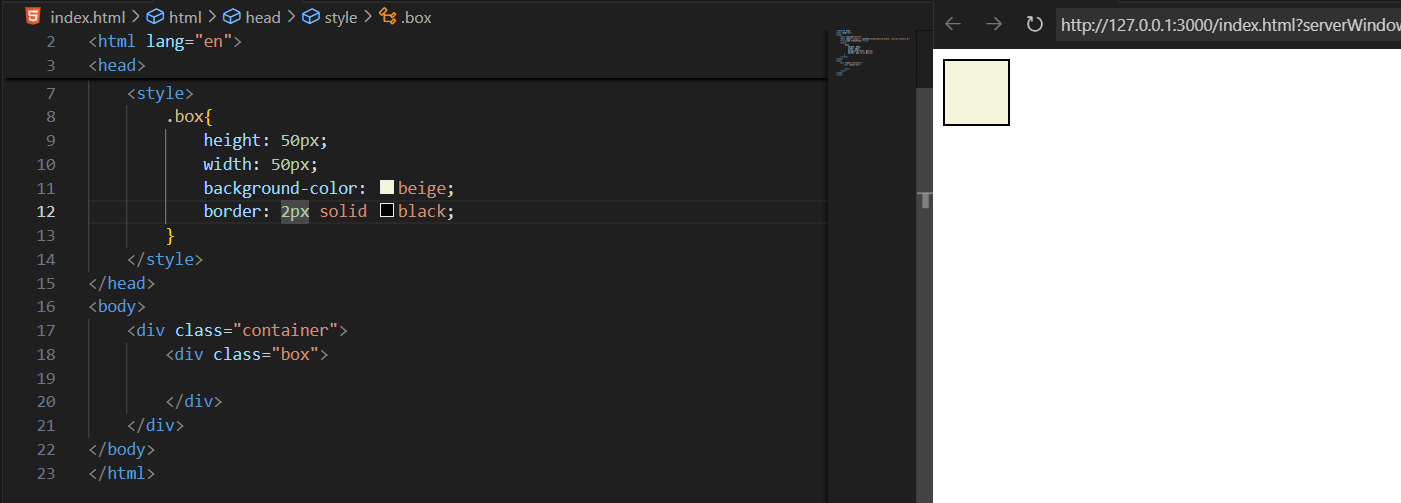
*transform: rotate(45deg);*

*}*

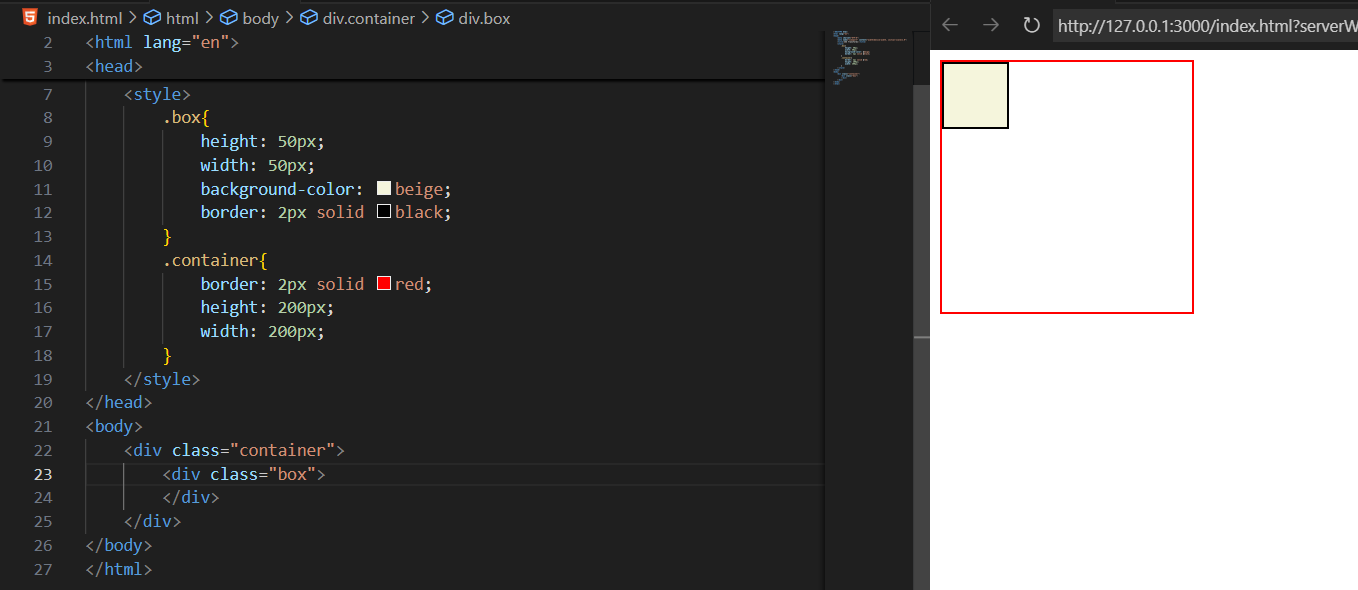
**Common Transform Functions:**

| **Function** | **Description** | **Example** |
| --- | --- | --- |
| translate(x, y) | Moves (shifts) the element along X & Y axes | transform: translate(50px, 20px); |
| rotate(angle) | Rotates the element clockwise or counterclockwise | transform: rotate(45deg); |
| scale(x, y) | Scales (zooms) the element | transform: scale(1.5, 1.5); |
| skew(x, y) | Skews (tilts) the element | transform: skew(20deg, 10deg); |
| matrix(a, b, c, d, e, f) | Advanced 2D combination of all transforms | Rarely used manually |

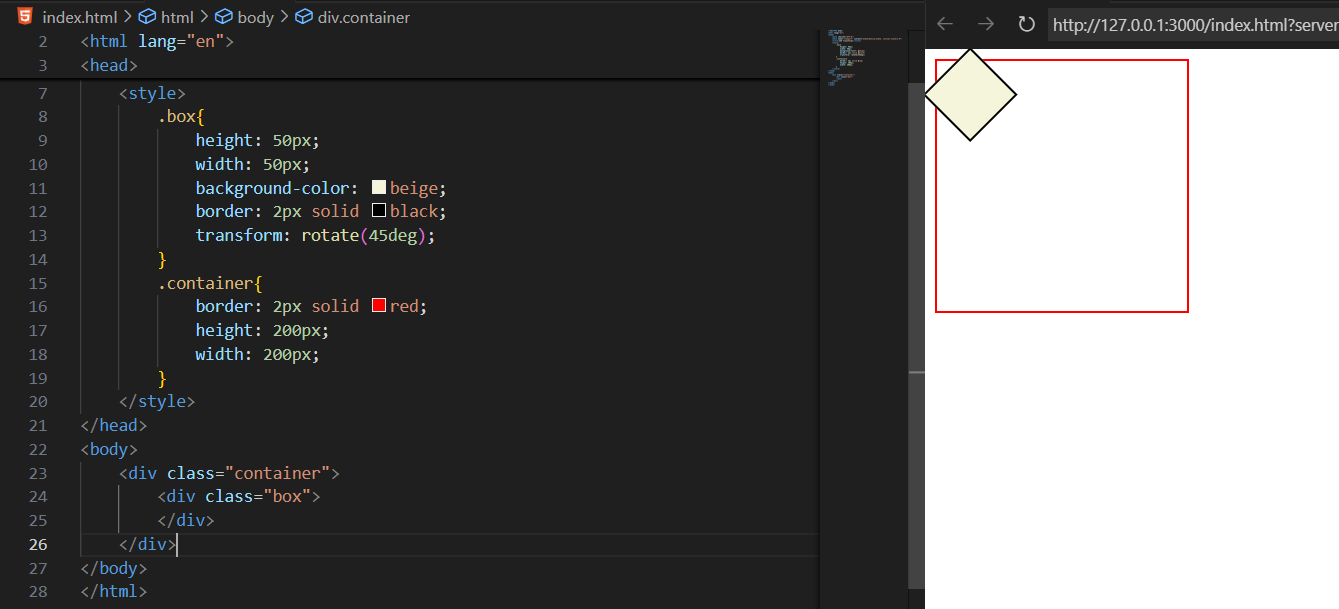
A basic example without any transforms:



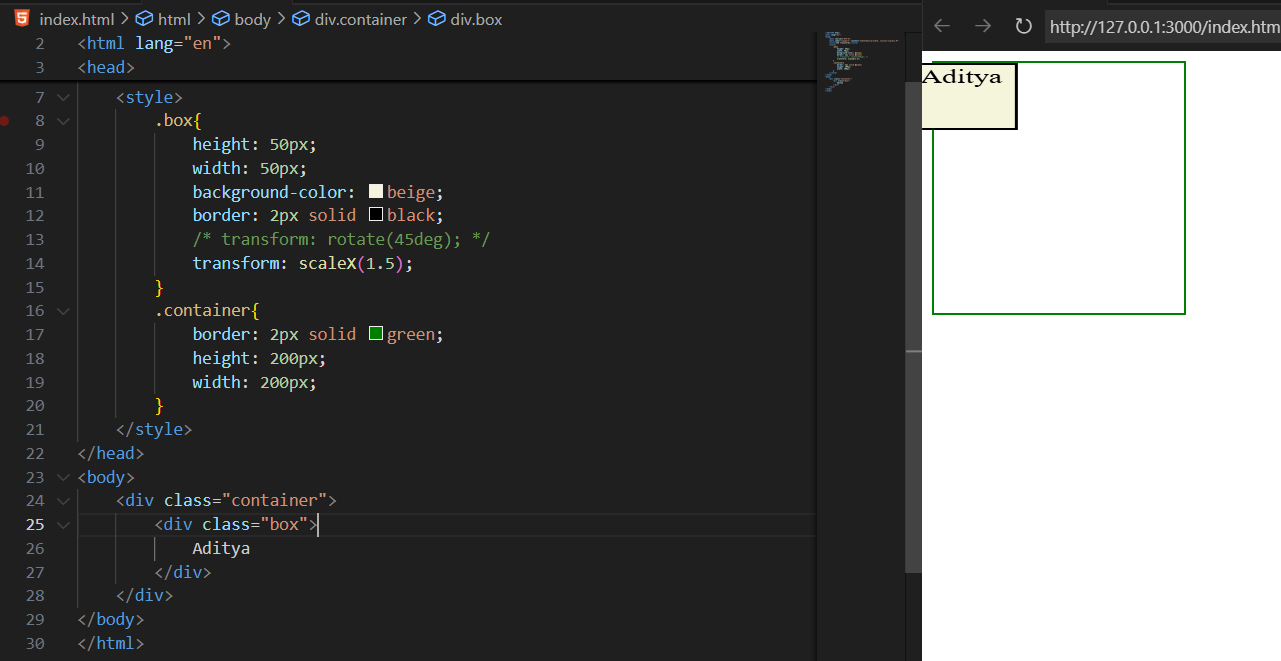
One more basic example: without transform



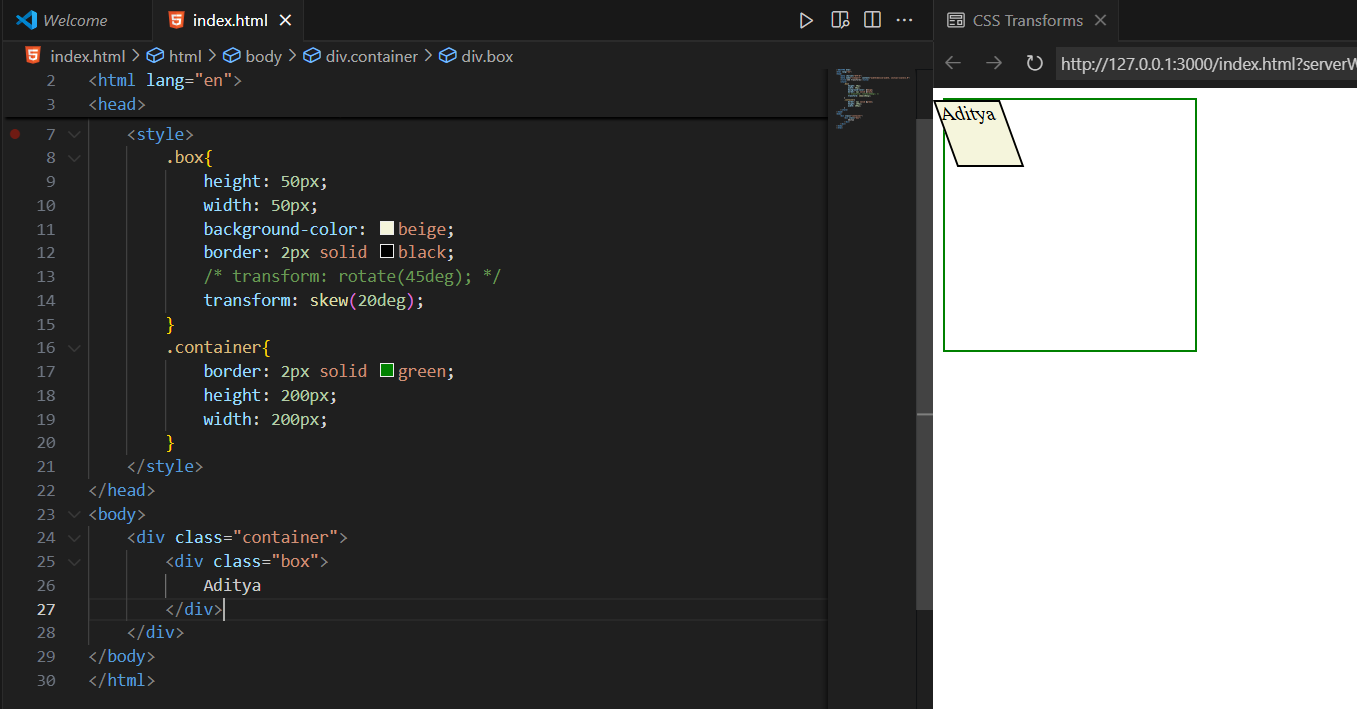
Now, applying the transform:rotate(45deg):



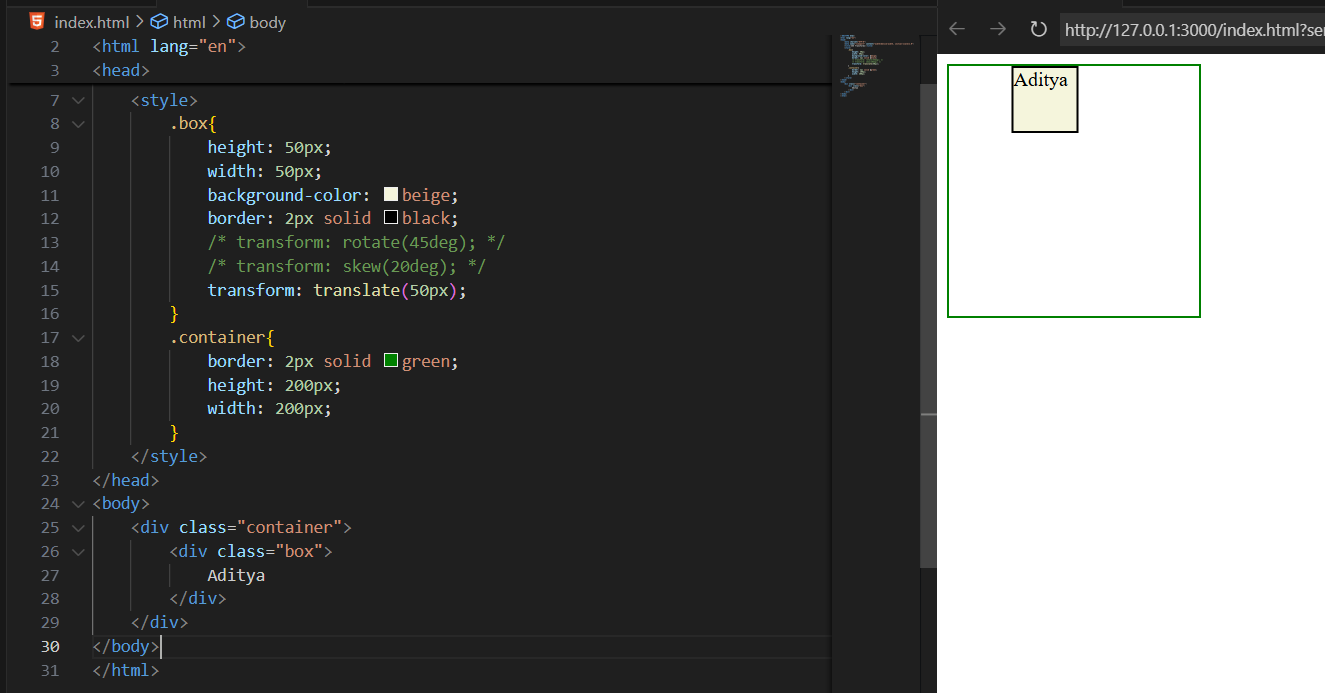
Example: we used scale() in order to scale the element



Example: skew()



Example: translate()



**CSS Transitions:**

**What is a CSS Transition?**

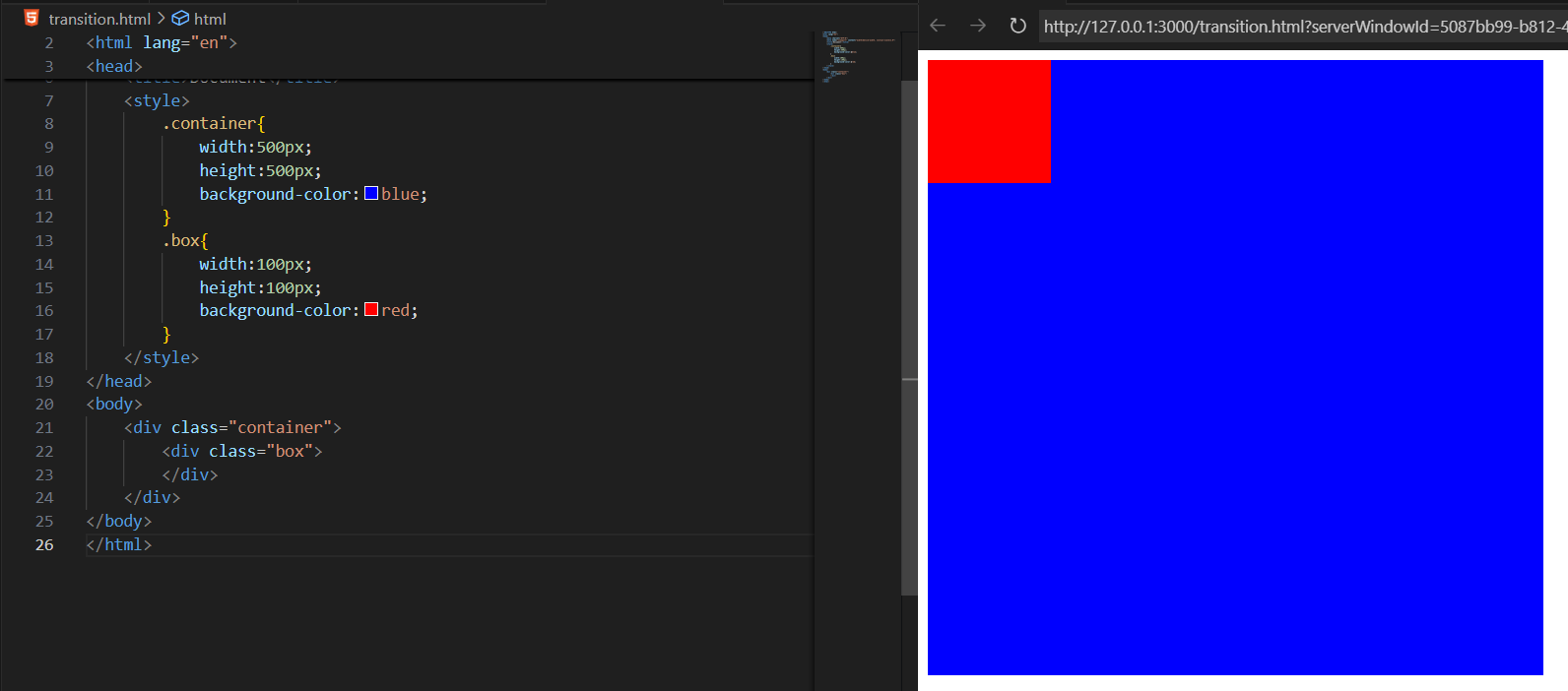
A transition lets you gradually change a CSS property’s value over time — instead of changing it instantly.

Basic syntax:

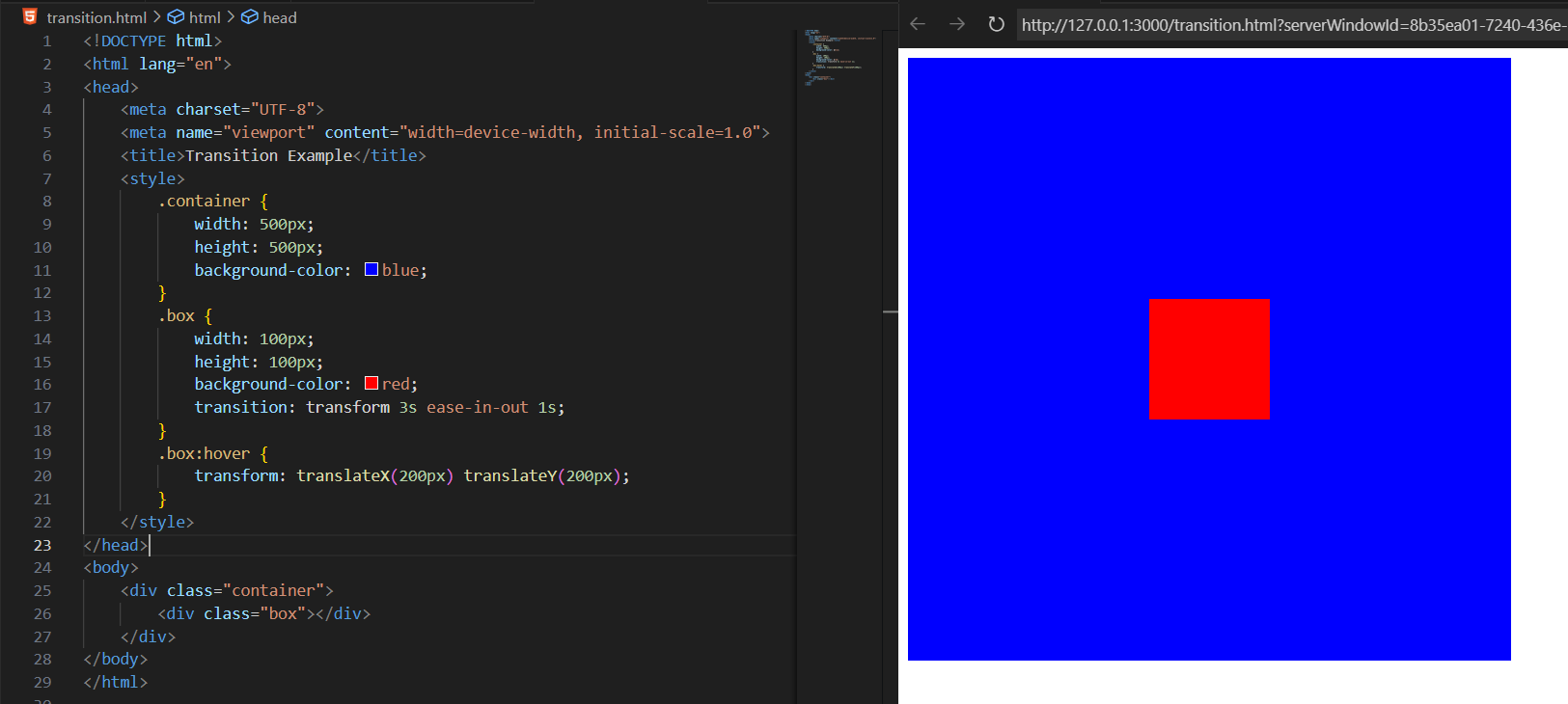
transition: property duration timing-function delay;

| **Part** | **Meaning** | **Example** |
| --- | --- | --- |
| **property** | CSS property to animate | width, color, transform, etc. |
| **duration** | How long the change takes | 1s, 0.5s |
| **timing-function** | Speed curve of animation | ease, linear, ease-in, ease-out, ease-in-out |
| **delay** | When to start (optional) | 0s, 1s |

A very basic example: where two <div> without transition are there:



Now, applying the transition property: we will see that red box slowly, moves to 200px in both X and Y axis.



**CSS Animations:**

**What is a CSS Animation?**

CSS animations let you move, change, or transform elements automatically over time, without user interaction. Unlike transitions, animations don’t need hover or click — they can start on page load, repeat, and have multiple stages.

**Basic syntax:**

*selector {*

*animation-name: myAnimation;*

*animation-duration: 2s;*

*animation-timing-function: ease-in-out;*

*animation-delay: 0s;*

*animation-iteration-count: infinite;*

*animation-direction: alternate;*

*}*

**Then define the animation using @keyframes 👇**

*@keyframes myAnimation {*

*from { background-color: red; }*

*to { background-color: blue; }*

*}*

**Or with percentage stages:**

*@keyframes myAnimation {*

*0% { transform: translateX(0); }*

*50% { transform: translateX(100px); }*

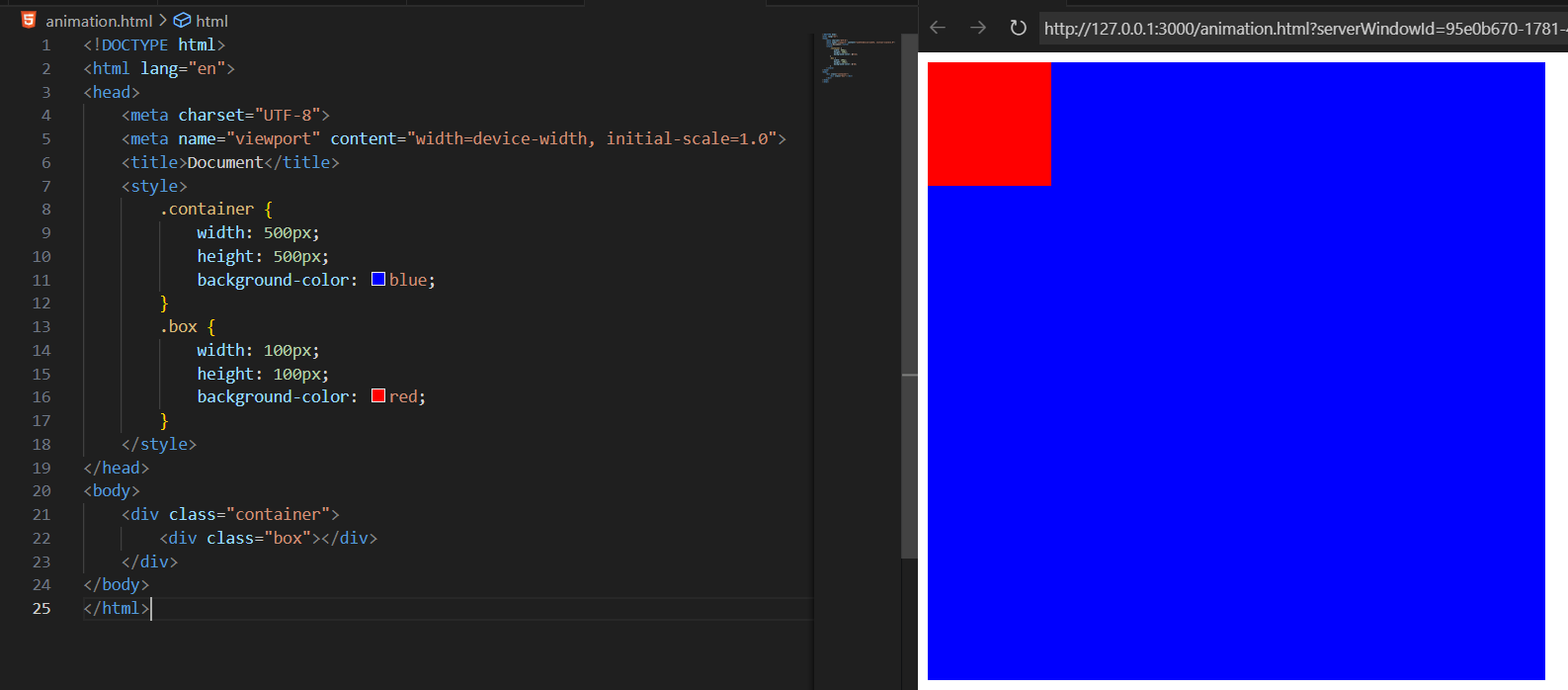
*100% { transform: translateX(0); }*

*}*

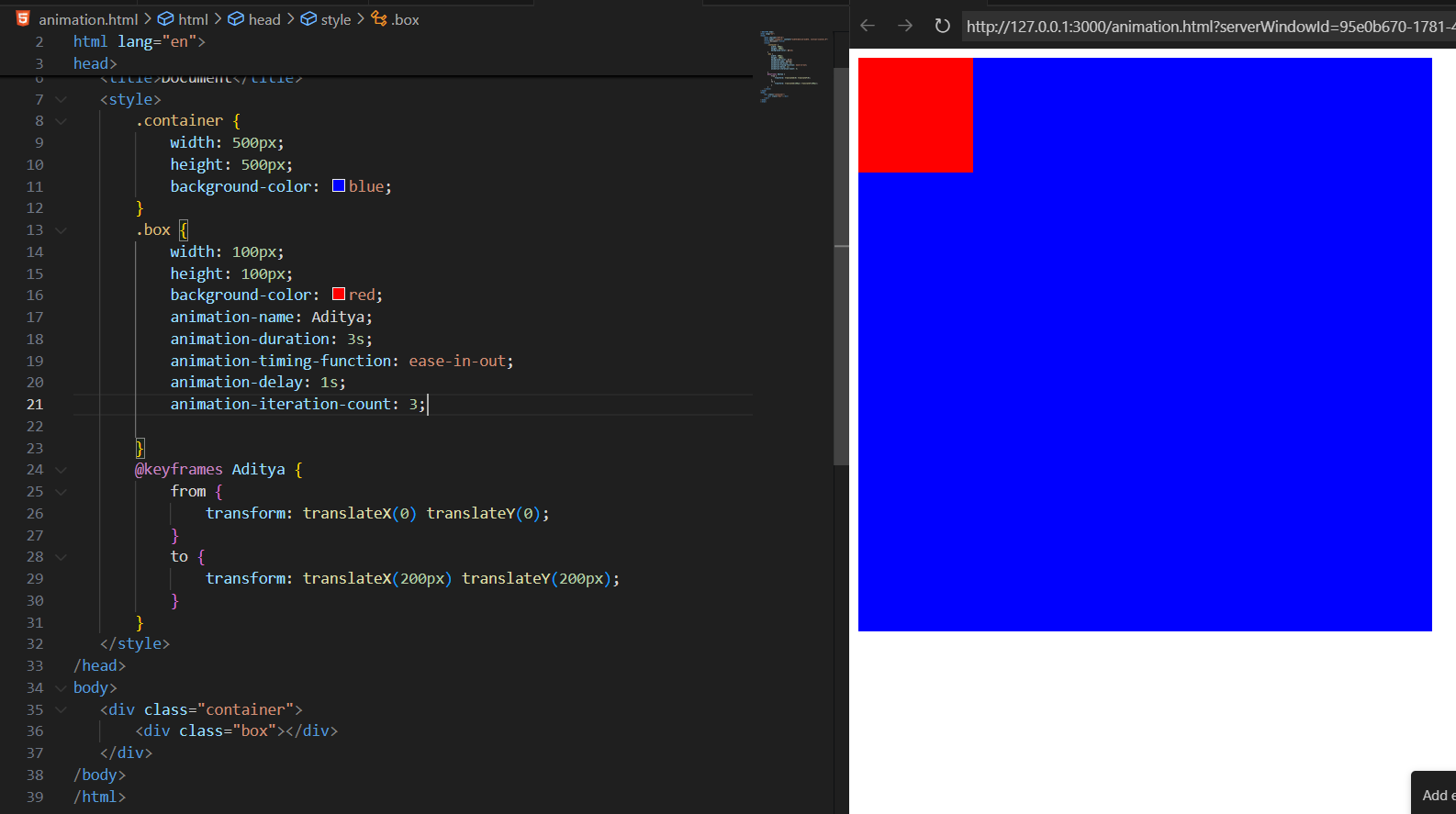
Main animation properties:

| **Property** | **Description** | **Example** |
| --- | --- | --- |
| animation-name | Name of @keyframes | slide |
| animation-duration | How long it lasts | 2s |
| animation-timing-function | Speed curve | ease, linear, etc. |
| animation-delay | Wait before start | 1s |
| animation-iteration-count | How many times | infinite, 3 |
| animation-direction | normal, reverse, alternate | alternate = forward then backward |
| animation-fill-mode | Keeps end state | forwards |
| animation-play-state | running or paused | pause/resume control |

A basic example without any animation:



Now, adding the animation:



--The End--