CHANGE & MOTION

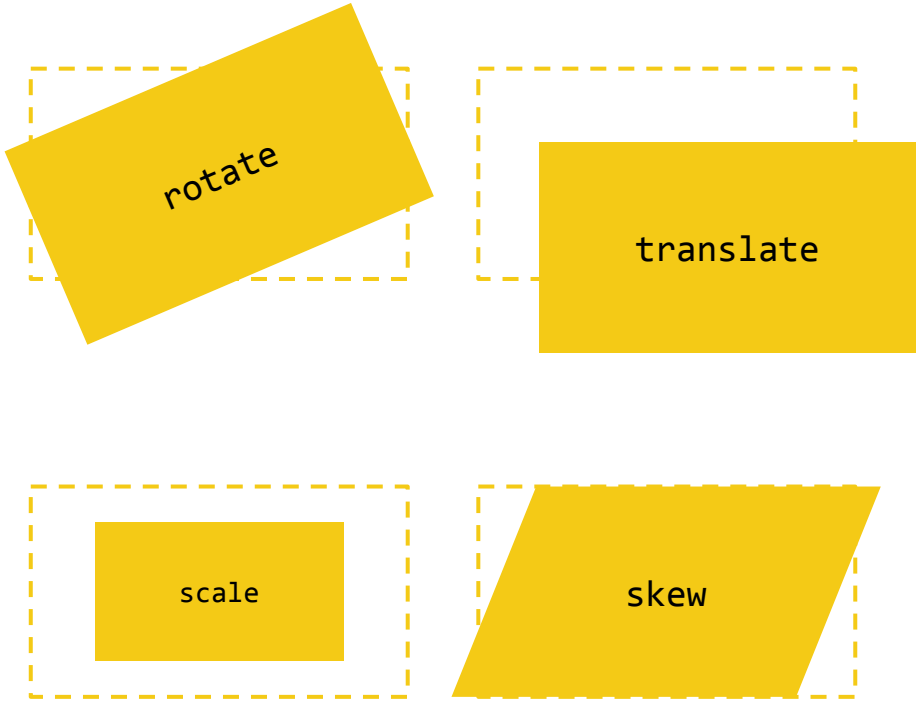
CSS transitions and animations

Lecture 6: Jan. 19 2016

CSS Transforms

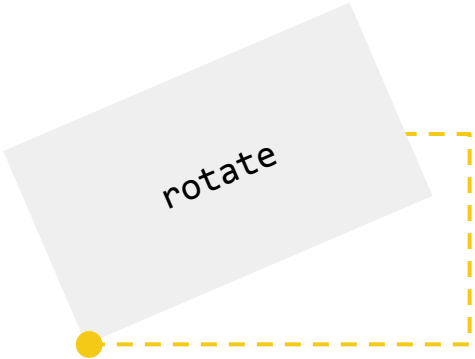
Manipulating elements

CSS 2D Transforms



CSS transform allows you to change the shape and position of HTML elements without disrupting the normal flow.

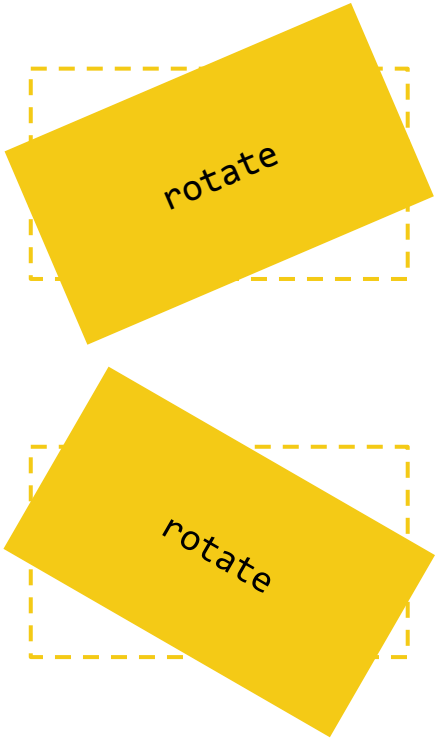
Transform Origin



`transform-origin` specifies the origin point of the transformation performed. This can be specified in multiple forms, including keywords, % values, and px. It is at the center of the element by default.

```
.example {
     transform: rotate(-30deg);
     transform-origin: bottom left;
}
```

Rotate



rotate() Rotates the div clockwise (+) or counter-clockwise (-), specified in degrees (deg).

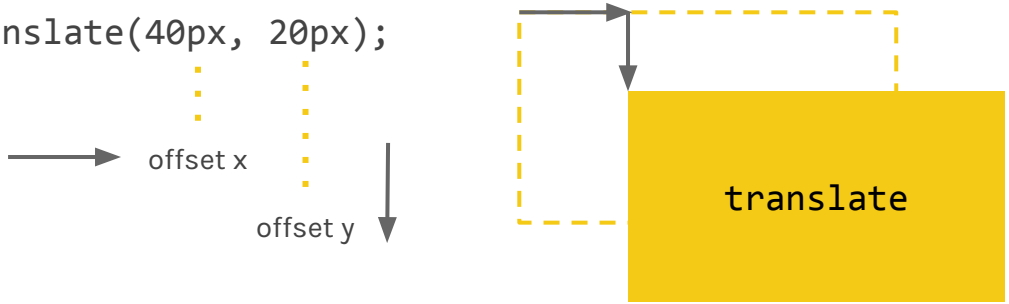
```
.example {
     transform: rotate(-30deg);
}

.example {
     transform: rotate(30deg);
}
```

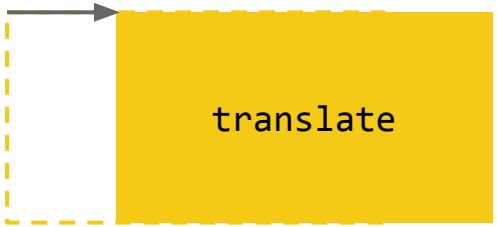
Translate

translate() moves an element sideways, up, or down. This can be specified in any length unit.

```
.example{
     transform: translate(40px, 20px);
}
```



Translate on one axis




```
.example {
     transform: translateX(30px);
}
```

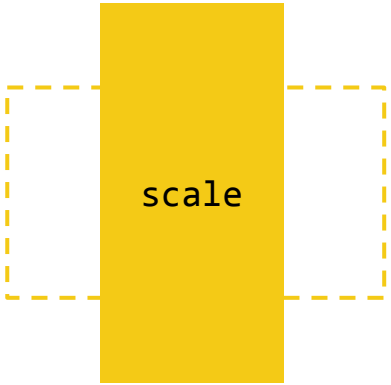

```
.example {
     transform: translateY(20px);
}
```

Scale

scale() stretches an element horizontally and/or vertically. Scale values are unitless. This also applies to the font-size, padding, height, and width of an element.

```
.example {
     transform: scale (.7);
}
```

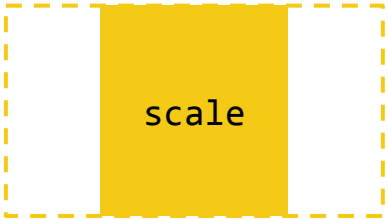

```
.example {
     transform: scale(.5, 1.5);
}
```



scale x

scale y

Scale on one axis



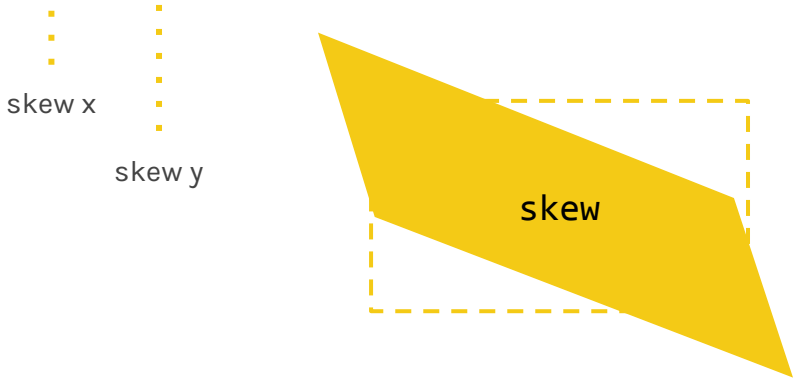

```
.example {
    transform: scaleX(.5);
}
```


```
.example {
    transform: scaleY(.5);
}
```

Skew

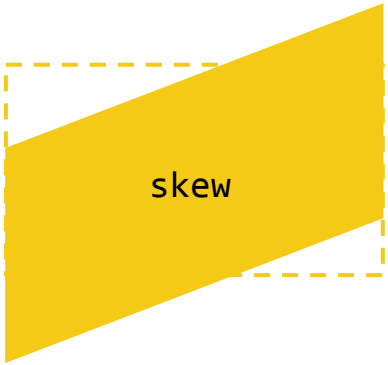
skew() stretches an element horizontally and/or vertically. Skews are defined in degrees (+, -). Contained elements, such as text, will also be skewed.

```
.example {
     transform: skew(10deg,30deg);
}
```



Skew


```
.example {
     transform: skewX(-20deg);
}
```




```
.example {
     transform: skewY(-20deg);
}
```

CSS Transitions

Easing changes in CSS

Combining Transforms

```
.example {
    transform: scale(.7, 1.5) rotate(30deg) skewY(-15deg) translate(200px, 20%);
}
```

- Multiple transforms can be applied to the same element with a space in between
    - Note: you cannot declare transforms separately; the latter will override the former.
- Transforms are applied in the order they are declared
- More on transforms including perspectival / 3D effects

CSS Transitions

```
.button {
     color: white;
     background-color: #f4ca16;
     transition: background-color 0.3s ease-in
     0.2s;
}

.button:hover {
     background-color: tomato;
}
```

Transitions allow property changes in CSS values to occur smoothly over a specified duration.

Transitions requires a trigger (such as : hover) to take effect. They are declared to the element targeted for the change.

Specifying Transitions

```
transition: background-color 0.3s ease-in 0.2s;
```

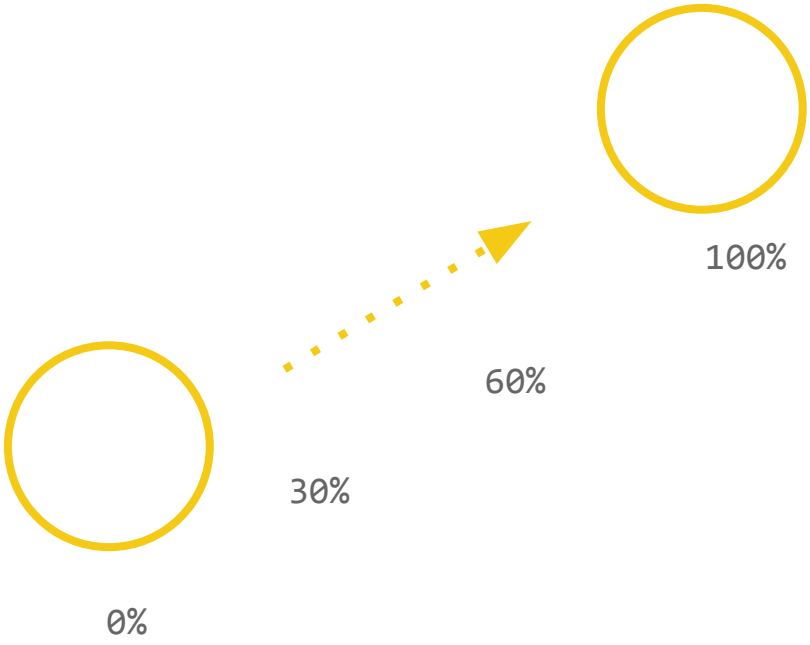
shorthand for

syntax	description	possible values
`transition-property: background-color;`	property being transitioned (or use `transition-property: all`)	See a list of CSS properties that can be transitioned.
`transition-duration: 0.3s;`	duration of effect	seconds (s) or milliseconds (ms)
`transition-timing-function: ease-in;`	the transition style	See common easing effects
`transition-delay: 0.2s;`	delay until starting effect	seconds (s) or milliseconds (ms)

CSS Animations

Setting elements in motion

Animations

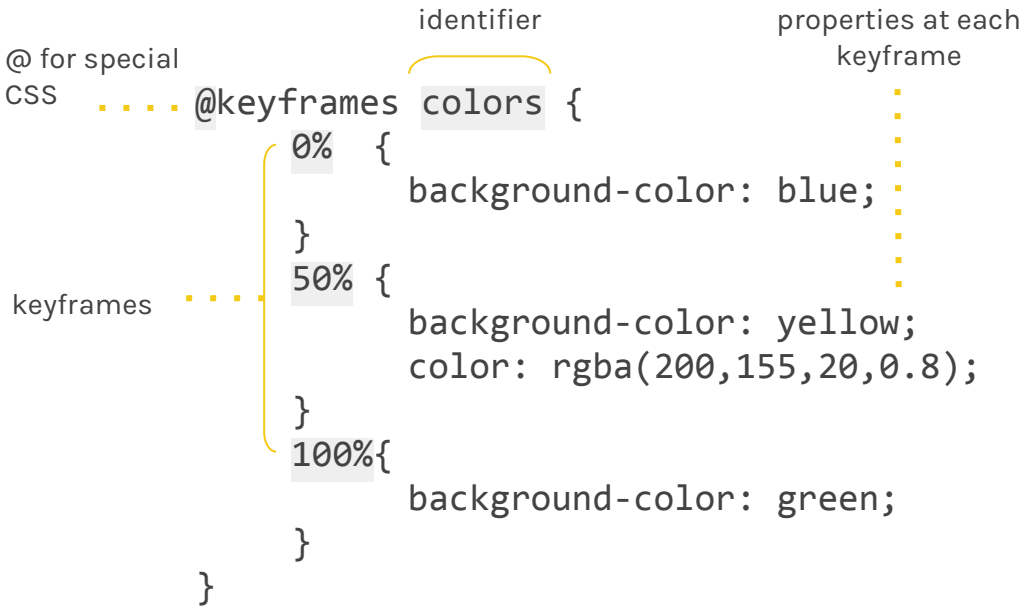


Using just CSS, you can create animation sequences for any element. (Animations allow motion without triggers.)

Animations consist of two separate sets of declarations:

- **@keyframes**: specifies the state of the element at a certain time point (relative to defined timing of animation.)
- **animation properties** apply the keyframe animation to one or many elements

Keyframes



- **identifier**
  - chosen name for the animation. This must match the name used to declare animation properties
- **keyframes**
  - timestamps / waypoints in the animation
  - `from, to, %`

Animation properties

target element

```
p {
    animation: colors 3.5s linear 0.2s 3 alternate;
}
```

shorthand for

syntax	description	possible values
`animation-name: colors;`	identifier given to animation in @keyframes declarations	any name without spaces
`animation-duration: 3.5s;`	animation duration	seconds (s) or milliseconds (ms)
`animation-timing-function: ease-in;`	animation style	same as transitions
`animation-delay: 0.2s;`	delay until starting animation	seconds (s) or milliseconds (ms)

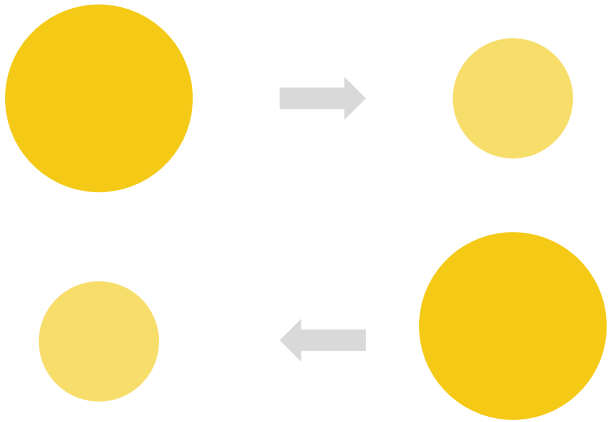
Animation properties

```
animation: colors 3.5s linear 0.2s 3 alternate;
```

shorthand for

syntax	description	possible values
`animation-iteration-count: 3;`	number of times the animation runs	infinite or numbers
`animation-direction: alternate;`	from what direction the animation begins	`normal, reverse, alternate alternate-reverse`
`animation-play-state: running;`	whether to play or pause	`running` (default), `paused`
`animation-fill-mode: none;`	whether to apply styles before and after the animation executes	`none` (default), `forwards, backwards, both`

Combine with Transform



Transitions and animations can also be combined with any of these CSS properties, but we should be careful as some properties may eat up some of your performance (and your animation may appear choppy, Transform functions, however, such as translate() scale() and rotate() are safe to use.

See example.

Change & Motion: Review

You should now have an understanding of how to:

- ❏ Transform elements
- ❏ Transition CSS changes and transforms
- ❏ Animate CSS transforms