Transitions and Animations

Transition

- Shows CSS property change over time
    - Color/opacity/size/position/etc
- Limited control over speed
- Runs to completion and is done

Animation

- Defines CSS property changes over time
    - Color/opacity/size/position/etc
- Good control over speed
- Run once, N times, or infinite

Animation Accessibility

Animations (with or without transitions) can be great

- Reduce change blindness
- "feel smooth"

But some people are sensitive to motion

- Giving someone a migraine is not good

All motion should be placed in media queries

- Confirm `prefers-reduced-motion` is `no-preference`

I omit in slides/notes for space

Transitions

When a CSS property changes…

- Opening a menu
- Expanding content

…you can slow down and show the change

Transition Syntax

- `transition-property` - CSS prop(s) to animate
- `transition-duration` - how long transition takes
- `transition-timing-function` - fast/slow start/end?
- `transition-delay` - pause before starting

Or `transition` shorthand to take all 4

- Minimum of property and duration

Placed in "base" selector, not "changed"

transition-property

```
transition-property: color;
```

- Comma-separated list of properties to animate
- Or `all`, but avoid using `all`
    - Performance hit
    - New properties could animate in the future

transition-duration

```
transition-duration: 1s;
```

- How long to animate to completion
- For interaction, don't be too slow
    - Humans get impatient in less than 1s

1s 2s

transition-timing-function

```
transition-timing-function: ease-in-out;
```

- Speed of progress of animation
  - `linear` - Advance steadily
  - `ease` - (default) Middle faster than start/end
  - `ease-in` - Slower start, speeds up
  - `ease-out` - Starts fast, slows down
  - `ease-in-out` - Like `ease`, but slower start/end
  - `cubic-bezier()` - Define with MATHS
  - A few others

transition-delay

```
transition-delay: 250ms;
```

- Delay before starting transition
- Why?
    - Different property transitions
    - Prevent "flyby" hover effects

delay 0

delay 250ms

transition

- Common to accept defaults
  - `-property` and `-duration` lack defaults

    ```
    transition: background-color 1s;
    ```

- Setting all 4 properties

  ```
  transition: color 1s ease-in-out 250ms;
  ```

- Setting multiple properties

  ```
  transition: width 1s ease, height 1s ease;
  ```

- Alternative way to set multiple properties

  ```
  transition: 1s ease;
  transition-property: width, height;
  ```

- Remember `prefers-reduced-motion`!

Applying Transitions

Examples will often use `:hover`

Actual usage *might* involve `:hover`

- And `focus`, `focus-within`

More often uses JS to apply/remove a class

- Element now matches different selectors
- Triggers animation for changed properties

A few other ways, but these are most common

CSS Animation overview

- Transitions are "animations", not `CSS Animations`

CSS Animations

- **Define** CSS property changes over time
    - Color/opacity/size/position/etc
    - Transitions only show from existing rules
- Good control over speed
    - `transition-timing-function` is limited
- Run once, N times, or infinite
    - Transitions stop at the new state
    - Animations can repeat, reverse, or loop

Animation Syntax

Two parts:

- `animation` related CSS properties
    - similar to `transition`
    - more control
- `@keyframes` definition
    - defines behavior at different points of animation

Transitions animate between two states

Animations define states to animate between

Simple Animation Property

```
.target1 {
  display: inline-block;
  width: 3rem;
  height: 3rem;
  background-color: green;
}

.container1:hover .target1 {
  animation-name: bounce; /* our defined keyframe (coming)*/
  animation-duration: 2s;
  animation-direction: alternate;
  animation-iteration-count: infinite;
  /* animation: bounce 2s infinite alternate; */
}
```

Simple Animation Keyframe

```css
.container1:hover .target1 {
  animation: bounce 2s infinite alternate;
}

@keyframes bounce {
  from {  /* starting values */
    margin-left: 0%;
  }

  50% {
    background-color: red; /* changing it up! */
  }

  to {
    margin-left: calc( 100% - 3rem); /* end values */
  }
}
```

Animation Property Details

- Similar to `transition`
    - Has `-duration`, `-timing-function`, `-delay`
- `animation-name`: Name of keyframe definition
- `animation-iteration-count`: Number or `infinite`
- `animation-direction`: `normal`, `reverse`, `alternate`, `alternate-reverse`
- `animation-fill-values`: Styles w/not animating
    - `none`: (default) As if no animation
    - Other values: Use styles from a keyframe
- `animation-play-state`: `running` or `paused`

Keyframe details

- `from` and `to` required
- No inheritance from other `@keyframes`
- Per-property if step is repeated
- Animation is "smart" w/missing properties

```
@keyframes bounce { /* name is up to you! */
  from {
    margin-left: 0%;
  }

  50% {
    background-color: red; /* what do you expect? */
  }

  to {
    margin-left: calc( 100% - 3rem);
  }
}
```

Applying Animations

Similar to transitions:

- Always use `prefers-reduced-motion` media query
- Examples will use `:hover`
- Actual usage *might* use `:hover`/`:focus`/etc
    - Might animate on page load
    - Might be when JS add/removes a class
        - Triggers animation directly
        - Transitions triggered by *property* change

Animations in the Wild

https://president.northeastern.edu/#community

Notice how it "overshoots" and then backs up

Applying Transitions to our Dropdown

- Now using BEM class names
    - Not required, but a good demonstration
- `nav`, then `menu`
    - `menu` a block by itself
        - even though in `nav`
- `menu__item`, `menu__name`, `menu__submenu`
    - `menu__submenu` is *also* `submenu`
- Perhaps `submenu__link` if we need to style those?

Planning Reduced Motion

- Some users have vestibular disorders
- Some users just don't want a lot of motion
- Operating System has "reduced motion" setting
- Browser can read this setting
    - But nothing is automatic
    - We must consider it
- Can remove motion if set
    - Easy to make mistakes
- OR only apply motion if allowed
    - Easier to be sure

Transition or Animation?

- Transitions slow down a change in CSS properties
- Animations apply a set of CSS property changes

For Dropdown Navigation

- We have existing properties we are changing
    - Thus, we want a Transition

Transitions require steps to show

- `display:none` doesn't exist, can't show steps
    - `transition-behavior: allow-discrete;`
        - Displays immediately
        - Allows other properties to be seen
        - New! Very limited support

Height: auto doesn't have steps

Transitioning height from `0` to `auto`

- Doesn't work
- But `0` to a specific `height` works
- Because `auto` isn't a number (?!)
- Future: `calc-size` will do this

Problem: We can't really know what height to expect

What about a big max-height?

- `max-height` serves as a cap for `height`
  - But `height` doesn't get bigger than needed!
  - Can give `max-height` a safely "big" value

Transitioning `max-height` works!

- But...what if we use a big value?
- Notice the weird timing and delay
- Transition is still over full range of max-height

Thinking in another dimension

If we rotate on the X-axis 90 degrees...

- We see element "edge-on" (invisible)
- Hidden: `transform: rotateX(90deg);`
- Visible: `transform: rotateX(0deg);`

Almost there, but it grows and shrinks from middle

- `transform-origin: top;`

`rotate` vs `transform`

- Can use `rotate: x 90deg;`
- Transition on `rotate`
- But still `transform-origin`, not `rotate-origin`
- Check caniuse.com for `rotate`

Could also use `transform: scaleY()` or `scale: y`

Once working, use faster transition

- Users expect visible results within ~100ms (0.1s)
    - Completion should be within ~250ms (0.25s)
- It's called User Experience
    - Not Developer Showing Off Experience

Confirm Reduced Motion

Easy enough to change reduced motion setting

- And confirm it works
- Remember to change it back :)