CSS Transitions, and Animations

CSS can do what now?

CSS Transitions

What are they? Thank you for asking.

Transitions are a visual effect which allows a property to change it's CSS values, like those that occur on :hover or :focus. It can be used to make the changes happen smoothly over a specified duration – rather than happening instantaneously as is the normal behavior.

You know that cool effect of hovering over an image and something changes. An overlay appears to give you more info on the image or a menu opens up to add specific navigation to that image. Well these types of effect can be done with a bit of CSS.

CSS Transitions

Property	Value
background	Shorthand CSS property for setting the individual background values in a single place.
height	CSS property specifies the height of the content area of an element
opacity	CSS property specifies the transparency of an element, that is, the degree to which the background behind the element is overlaid.
top,right,left,bottom	CSS properties that lets you modify the coordinate space of an HTML element
width	CSS property specifies the width of the content area of an element.

CSS Transitions

Okay this sounds amazing!!! Give me knowledge…NOW!

Well first calm down. Second let’s talk about how it works. Transitions introduce a number of properties which together can be used to specify:

- The CSS property (or properties) to be transitioned
- The duration of the transition
- The timing function of the transition
- An optional delay.

CSS Transitions

Property	Value
transition-property	Specifies the name or names of the CSS properties to which transitions should be applied.
transition-duration	Specifies the duration over which transitions should occur.
transition-timing-function	Specifies a function to define how intermediate values for properties are computed.
transition-delay	Defines how long to wait between the time a property is changed and the transition actually begins.
transition	CSS Shorthand property to specify all of the above.

CSS Transitions

First Step: Transition Property

So to create a transition effect we need to specify which CSS property (or properties) the transition effect will be applied to first.

Syntax Check:
| = or
, = and
transition-property: none | all | [<IDENT>];
transition-property: background-color, height, width;

CSS Transitions

Second Step: Transition Duration
Now we need to get our transitions a duration which will determine how long the corresponding properties take to complete their transition. The transition-durations property accepts a comma-separated list of times, specified in seconds or milliseconds,

Syntax Check:

```
transition-duration: <time>;
transition-duration: 2s;
transition-duration: 2000ms, 4000ms;
```

CSS Transitions

Third Step: Transition Timing Function
We’re going to use this property to specify how the pace of the transition changes over its duration. This can be done in one of two ways. Either by specifying a number of pre-defined keywords, or by defining a custom timing function.

Syntax Check:
```
<timing-function> = ease | linear | ease-in | ease-out | ease-in-out;
<timing-function> = cubic-bezier(<number>, <number>, <number>,
<number>);
```

CSS Transitions

Last Step: Transition Delay
Finally we can create an optional delay using the transition-delay property. As with the transition-duration property, the transition-delay property accepts a comma-separated list of times, specified in seconds or milliseconds, which in this case determines the length of time between the transition being triggered and the transition starting.

Syntax Check:

```
transition-delay: <time>
transition-delay: 2s;
transition-delay: 2000ms, 4000ms;
transition-delay: -2s;
```

CSS Transitions

All together now: Transition
This shorthand property can be used in place of the individual properties described previously.

Syntax Check:
<transition> = <transition-property> <transition-duration> <transition-timing-function> <transition-delay>;
transition: background-color 6s ease 3s;

CSS Animations

CSS animations make it possible to animate transitions from one CSS style configuration to another.

Animations consist of two components, a style describing the CSS animation and a set of keyframes that indicate the start and end states of the animation's style, as well as possible intermediate waypoints.

CSS Animations

The three key advantages to CSS animations over JavaScript animation techniques:

1. They're easy to use for simple animations; you can create them without even having to know JavaScript.
2. The animations run well, even under moderate system load.
3. Letting the browser control the animation sequence lets the browser optimize performance and efficiency.

CSS Animations

KeyFrames and You!

The main component of CSS animations is @keyframes, the CSS rule where animation is created. Think of @keyframes as being stages along a timeline.

To make the CSS animations work, you'll need to bind the @keyframes to a selector i.e class,id, or tag. This will parse all the code inside the @keyframes declarations and change the initial style to the new style, based on the stages.

CSS Animations

KeyFrames and You!

Syntax Check

```
@keyframes Fade {
  0% {
    opacity: 1;
  }
  100% {
    opacity: 0;
  }
}
```

Syntax Check

```
@keyframes Fade {
  from {
    opacity: 1;
  }
  to {
    opacity: 0;
  }
}
```

Syntax Check

Shorthand version

```
@keyframes Fade {
  to {
    opacity: 0;
  }
}
```

CSS Animations

Property	Value
animation-name	@keyframes name aka Fade like our example.
animation-duration	Specifies the duration over which the animation should occur.
animation-timing-function	Specifies the animation speed.
animation-delay	Defines how long to wait before the animation starts.
animation-iteration-count	Defines how many times we will iterate through the animation.
animation-direction	Defines the ability to change the loop direction.
animation-fill-mode	Specifies which styles will be applied to the element when our animation is finished.

CSS Animation

Syntax Check:

```
.supercool-element {
  animation-name: Fade;
  animation-duration: 6s;
  animation-delay: 2s;
  animation-iteration-count: infinite;
  animation-timing-function: linear;
  animation-direction: alternate;
}
```

or

```
.supercool-element {
  animation: Fade 6s 2s infinite linear alternate;
}
```

GROUP LAB TIME

Using the codepen examples given earlier create your own transitions and animations.

Requirements:
Work as a team
Change more than one property of two different elements
Research the transform property and see if you can add to either your transition or animation elements