FEWD

FINAL PROJECTS

AGENDA

- Debugging
- Carousel Lab Pt. 2
- Advanced CSS Positioning
- Transitions
- Transformations
- Animations

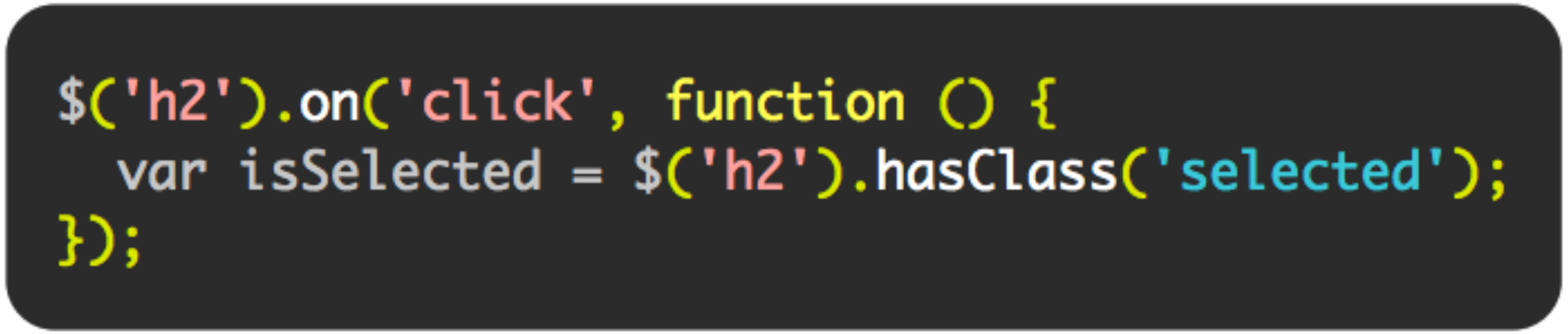
LEARNING OBJECTIVES

- Identify and differentiate between different CSS positioning techniques
- Familiarity with how animations and transitions can be used in CSS
- Understand how animation can still be controlled using JS
- Know the different ways to debug code and how to apply the concepts

REVIEW

HAS CLASS

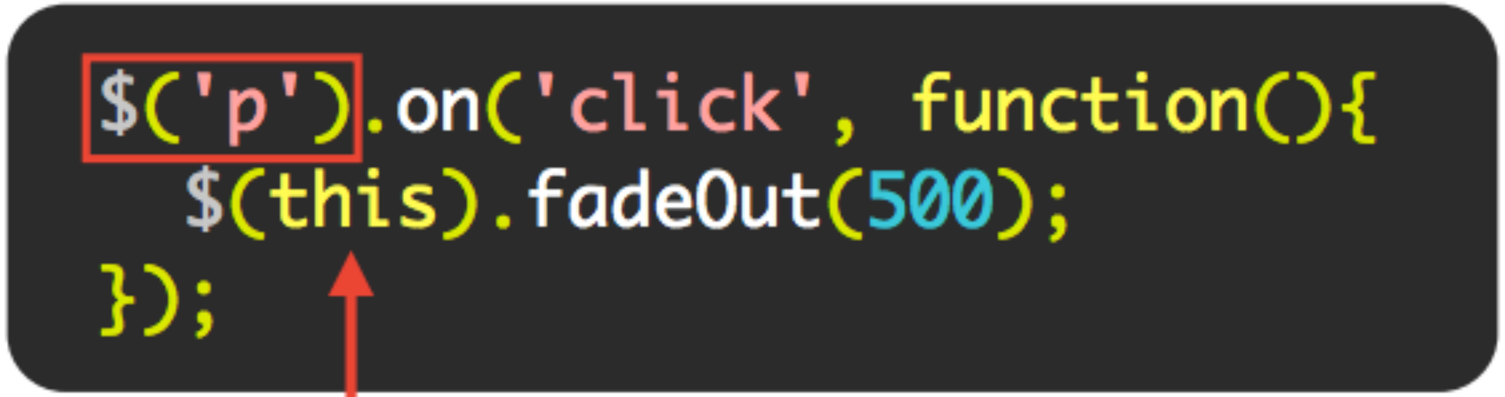
jQuery’s .hasClass() method is an easy way to tell whether or not an item is in a particular state.



```
$('h2').on('click', function () {
  var isSelected = $('h2').hasClass('selected');
});
```

THE KEYWORD ‘THIS’

this refers to whatever you *selected* with jQuery



```
$('p').on('click', function(){
  $(this).fadeOut(500);
});
```

Notice — no quotes around this!

OBJECTS

- OOP- Object oriented Programming
- Lets us write reusable code to keep track of data
- Objects have traits that are common to versions of itself
- These traits are called properties in javascript

Everything is an object!

BULLDOG AS AN OBJECT

Bulldog Properties

- Legs - 4
- Sound - “Bark”
- Food - “Dog Food"

DECLARING OBJECTS

```
function myObject(){

};
```

ASSIGNING PROPERTIES

```
function myObject(){
   this.property = value;
};
```

MAKING NEW OBJECTS

```
var newObject = new object();
```

GETTING OBJECT VALUES

```
var newObject = new object();

newObject.propertyName;
```

DEBUGGING

DEBUGGING

WHY ISN'T IT WORKING?

DEBUGGING — WHERE TO START

This will tell you where to start your hunt.

THE IMAGE IS NOT MOVING

NONE OF MY CODE WORKS

Find the code that makes the image move

** Syntax error, check console*

DEBUGGING

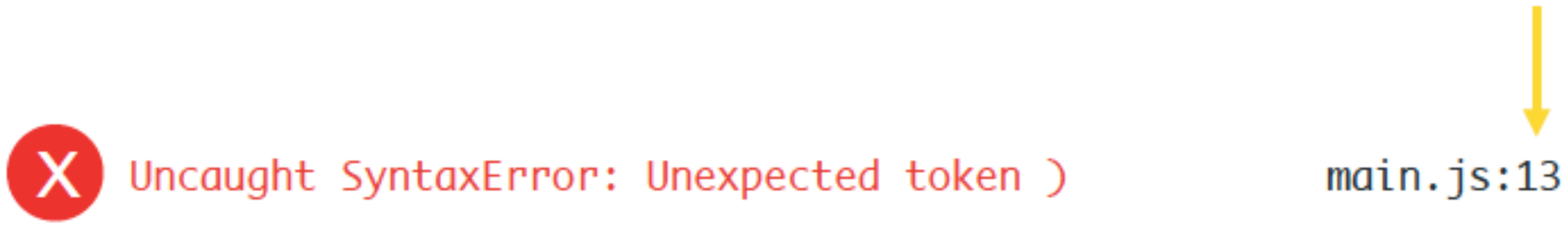
To access debugging console:

PC: CTRL+SHIFT+J
Mac: COMMAND+OPTION+J
Click the error

DEBUGGING — LEVEL 1

Check for errors in console

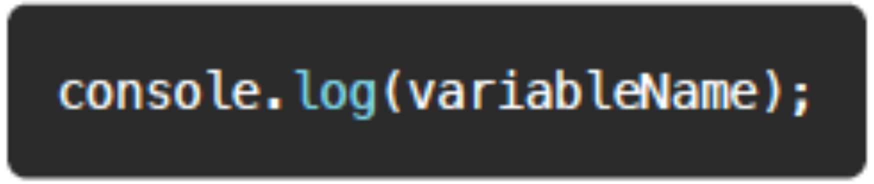
- The location may not be correct but is a good place to start.
- Ex: Unbalanced brackets or parentheses



```
Uncaught SyntaxError: Unexpected token )          main.js:13
```

DEBUGGING — LEVEL 2

- console.log() can be used to display variable values in the console. This is useful for debugging.



```
console.log(variableName);
```

This should print the element to the console.
If it doesn't, there's probably something wrong with your selector.

*Shortcut to access console: cmd + opt + j

DEBUGGING

Do some Googling!

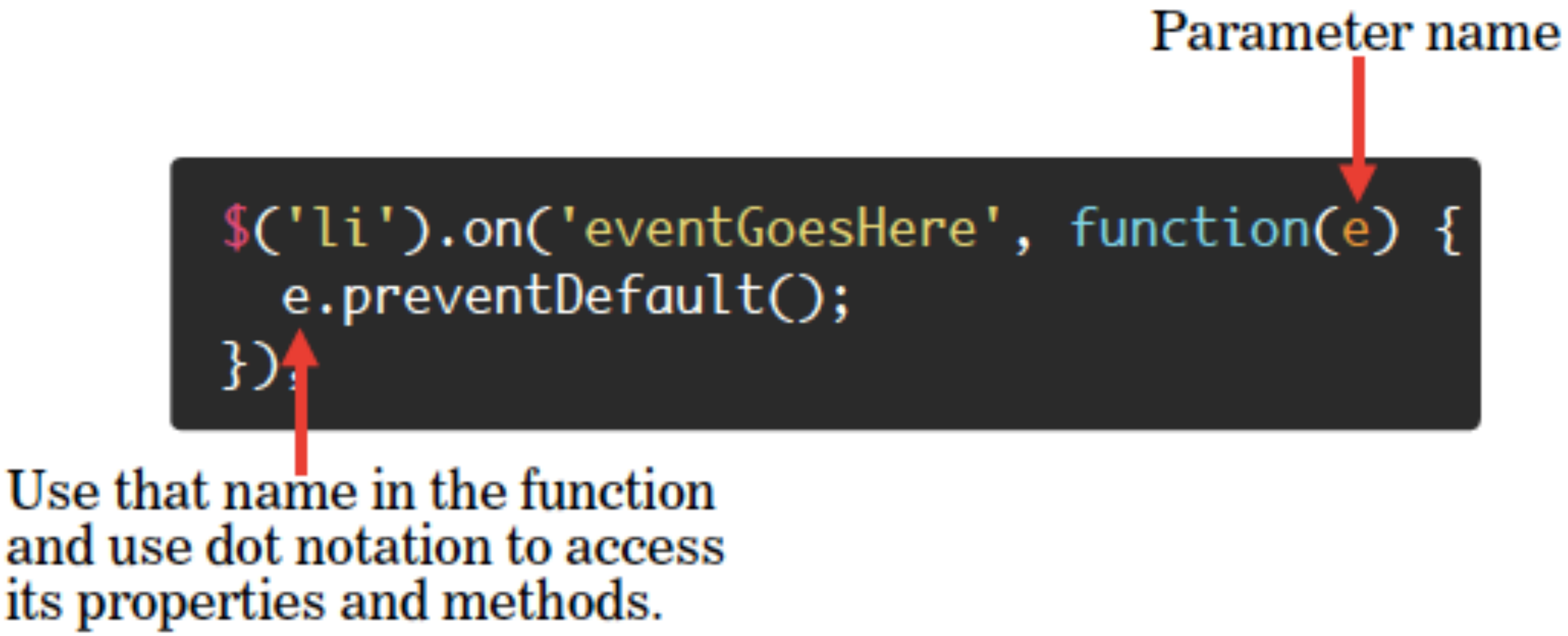
- Try Googling it
- Be ready to clearly articulate the problem (Write out what your problem is)

Use Slack!

PREVENT DEFAULT

JQUERY METHODS — THE EVENT OBJECT

- The event object has properties and methods that tell you more about the event that took place.
- By using the preventDefault method, the default action of the event will not be triggered.



```
$('li').on('eventGoesHere', function(e) {
  e.preventDefault();
});
```

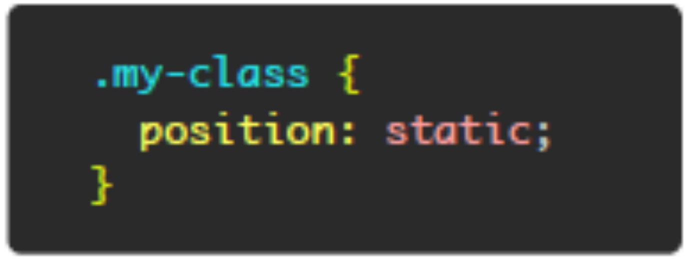
Parameter name
Use that name in the function and use dot notation to access its properties and methods.

CREATE EVENT LISTENERS

ADVANCED CSS POSITIONING REVIEW

STATIC POSITIONING

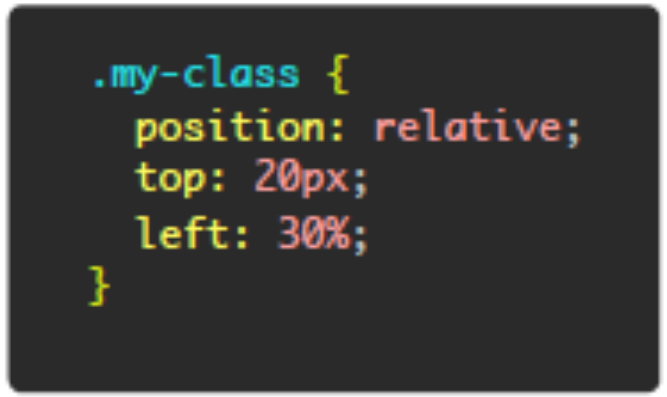
- This is the normal flow of the document, the **default**
- Elements render in order, as they appear in the document flow.



```
.my-class {
  position: static;
}
```

RELATIVE POSITIONING

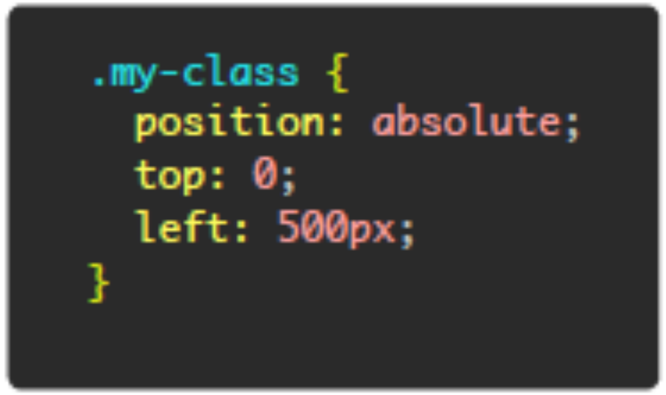
- Relative positioning moves an element *relative to where it would have been in normal flow.*
- For example, "left: 20px" adds 20px to an element's **left** position
- Creates a *coordinate system for child elements*.



```
.my-class {
  position: relative;
  top: 20px;
  left: 30%;
}
```

ABSOLUTE POSITIONING

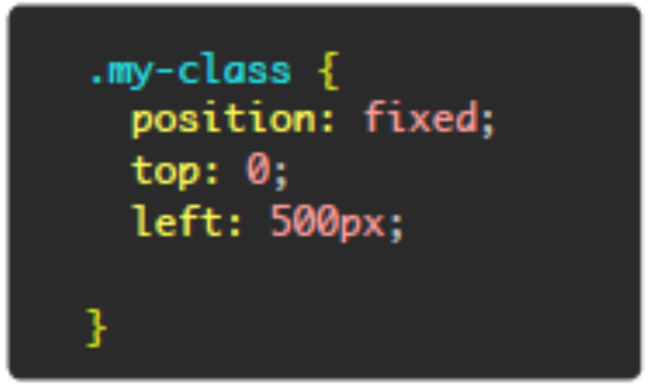
- When the *position* property is given a value of *absolute*, an element is taken out of the normal flow of the document.
- This element no longer affects the position of other elements on the page (they act like it's not there).
- You can add the *right*, *top*, *left* and *bottom* properties to specify where the element should appear relative to its first positioned (not static) ancestor element



```
.my-class {
  position: absolute;
  top: 0;
  left: 500px;
}
```

FIXED POSITIONING

- When the *position* property is given a value of *fixed*, the element is positioned in relation to *the browser window*
- When the user scrolls down the page, it stays in the same place.
- You can add the *right, top, left* and *bottom* properties to specify where the element should appear in relation to the browser window.

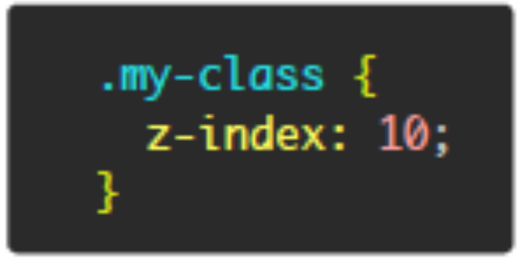


```
.my-class {
  position: fixed;
  top: 0;
  left: 500px;

}
```

OVERLAPPING ELEMENTS — Z-INDEX

- When using relative, fixed or absolute positioning, elements can overlap.
- When elements overlap, the elements that appear later in the HTML code sit on top of those that appear earlier in the page.
- If you want to control which elements are layered on top of each other, you can use the z-index property.
- This property takes a number — the higher the number the closer that element is to the front.
- Similar to 'bring to front' and 'send to back' in programs like *Adobe Illustrator*.



```
.my-class {
  z-index: 10;
}
```

WANT TO LEARN MORE?

Resources for more info/examples:

- A List Apart: CSS Positioning 101

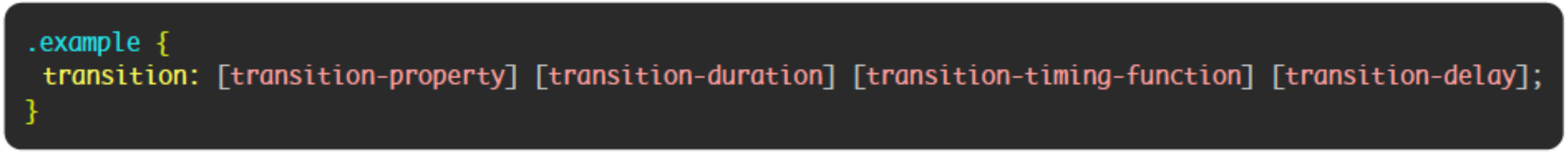
FEWD

CSS POSITIONING & ANIMATION

TRANSITIONS

TRANSITIONS

- Provide a way to control animation speed when changing properties
- Instead of having property changes take effect immediately, you can have them take place over a period of time.



```css
.example {
 transition: [transition-property] [transition-duration] [transition-timing-function] [transition-delay];
}
```

TRANSITIONS

PROPERTY

Which properties to animate

DURATION

How long the transition will last

TIMING FUNCTION

How the transition will run

TRANSITIONS - TRANSITION-PROPERTY

- Can specify a specific property to transition or "all" to transition all properties
- Default: all

```
div {
 transition: opacity 0.5s;
}
```

```
div {
 transition: all 0.5s;
}
```

```
div {
 transition: height 0.5s;
}
```

```
.example {
 transition: [transition-property] [transition-duration] [timing-function] [transition-delay];
}
```

TRANSITIONS - TRANSITION-DURATION

- A time value, defined in seconds or milliseconds

```css
div {
 transition: all 0.5s;
}
```

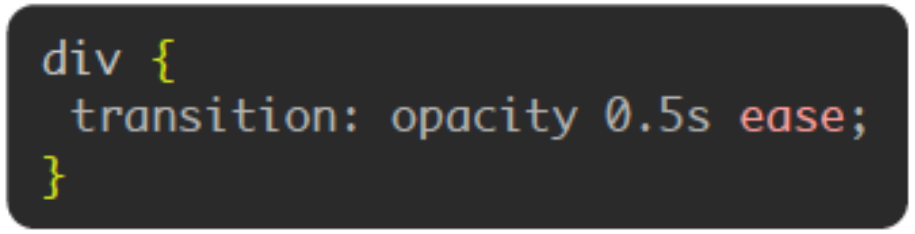
```css
div {
 transition: all 350ms;
}
```

```css
div {
 transition: all 3s;
}
```

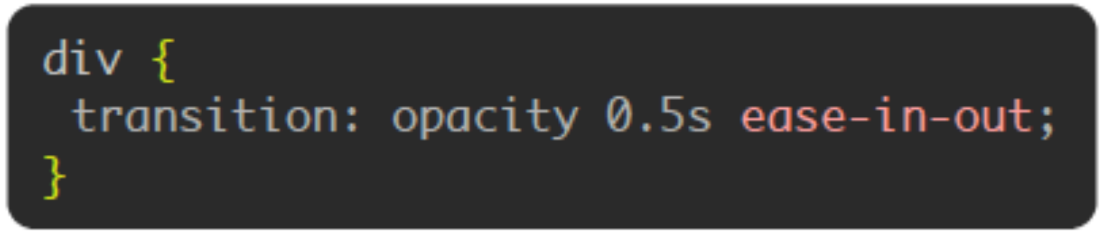
```css
.example {
 transition: [transition-property] [transition-duration] [timing-function] [transition-delay];
}
```

TRANSITIONS

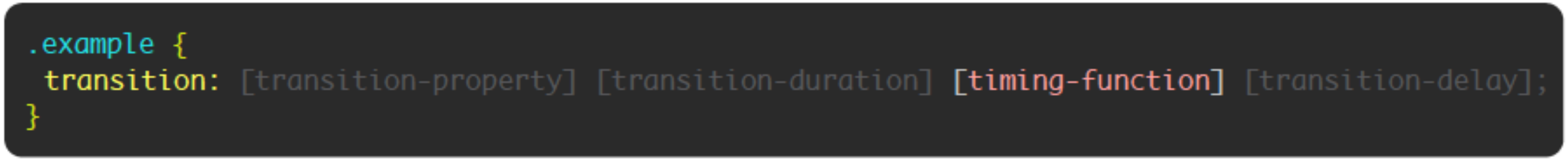
- Describes how a transition will proceed over its duration, allowing a transition to change speed during its course.
- Timing functions: ease, linear, ease-in, ease-out, ease-in-out



```
div {
 transition: opacity 0.5s ease;
}
```



```
div {
 transition: opacity 0.5s ease-in-out;
}
```

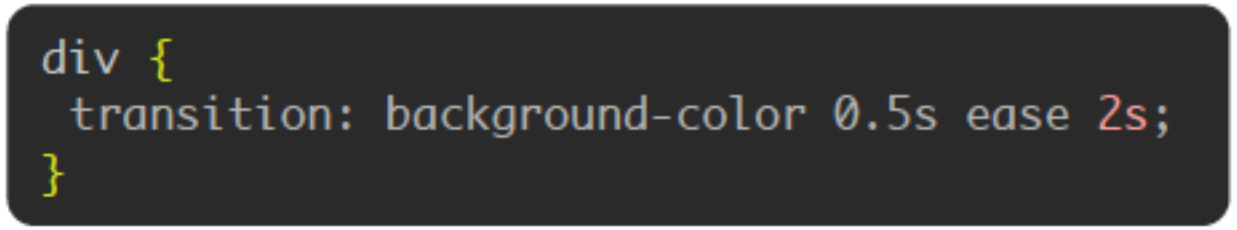


```
.example {
 transition: [transition-property] [transition-duration] [timing-function] [transition-delay];
}
```

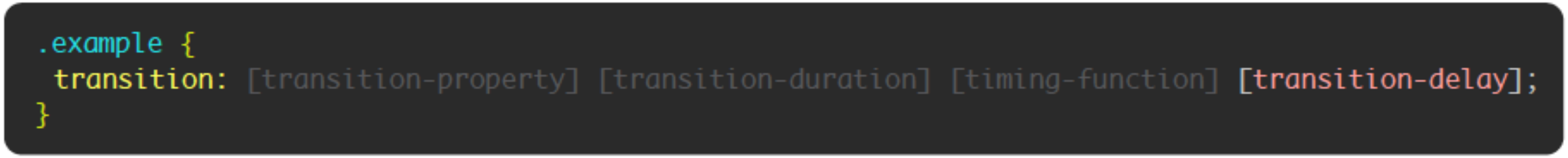
transition timing: W3 Schools

TRANSITIONS

- Length of time before the transition starts



```css
div {
 transition: background-color 0.5s ease 2s;
}
```



```css
.example {
 transition: [transition-property] [transition-duration] [timing-function] [transition-delay];
}
```

MORE FUN WITH TRANSITIONS — CODROPS

Fun CSS button styles: Creative buttons
Icon hover effects: Icon Hover Effects
Modal dialogue effects (advanced): Dialogue Effects

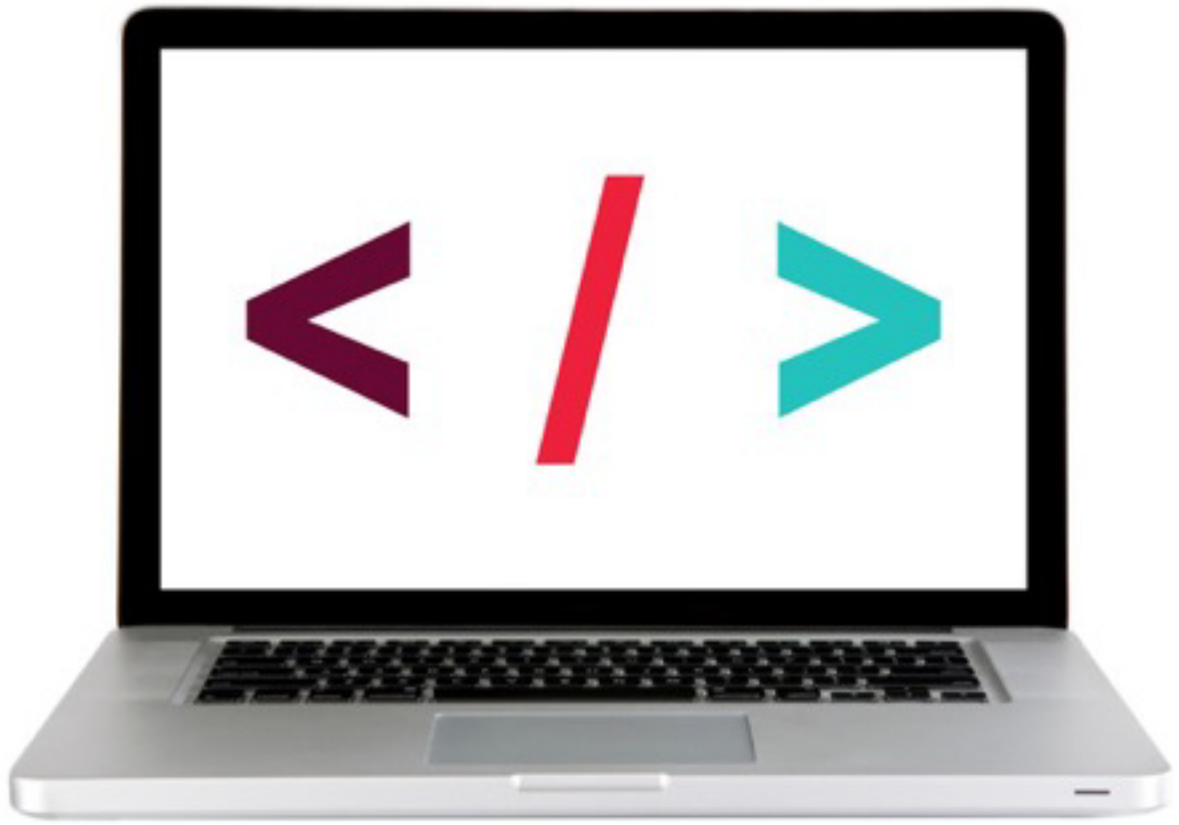
TRIGGERING TRANSITIONS

TRIGGERING TRANSITIONS

There are two ways to trigger CSS transitions:

1. Using the :hover CSS pseudo-class
2. Adding a class with jQuery

LET'S TAKE A CLOSER LOOK — TRIGGERING TRANSITIONS



ACTIVITY — BUTTON LAB

EXERCISE

KEY OBJECTIVE

- Practice using CSS transitions

TYPE OF EXERCISE

- Individual/Partner Lab

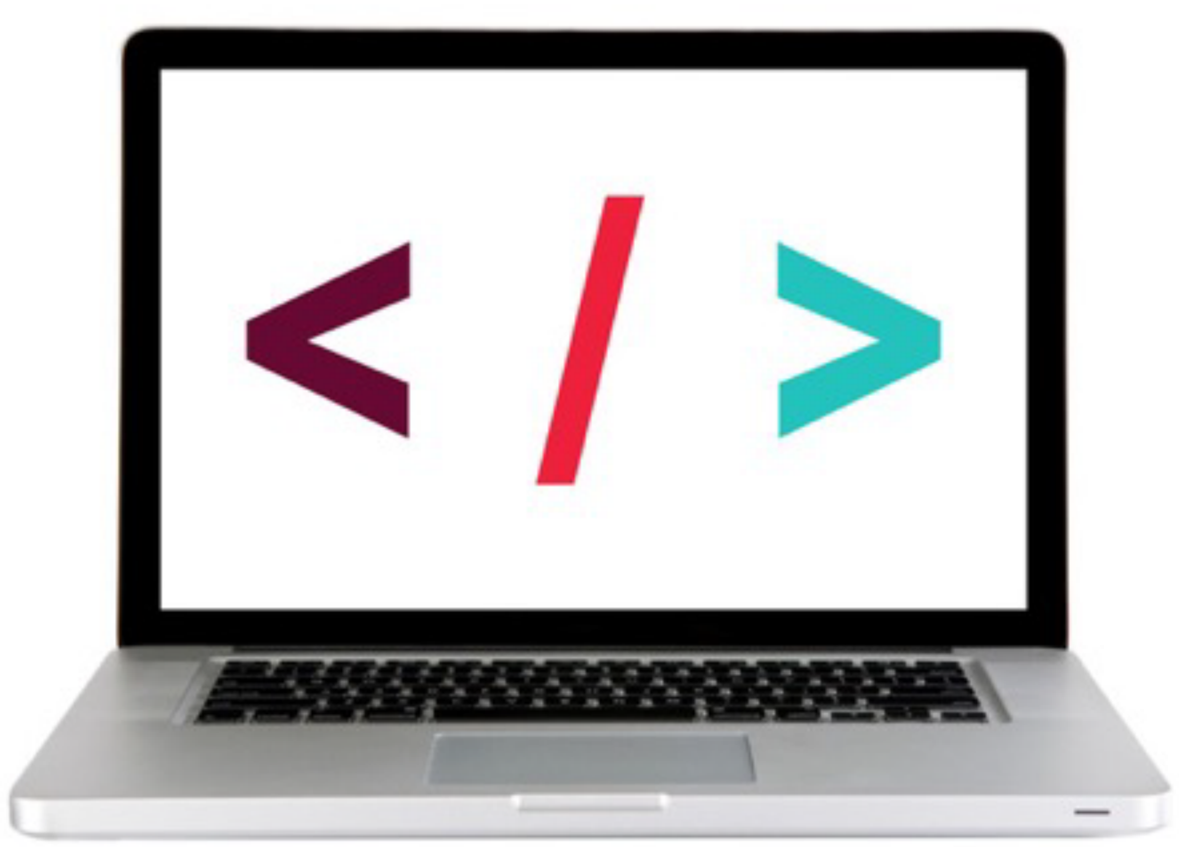
TIMING

6 min

1. Add :hover styles and transition to the button: **lesson11_starter_code > [2] transition_button_lab**

TRANSFORMATIONS

LET'S TAKE A CLOSER LOOK — TRANSFORM



transform: W3 Schools
transform-origin: W3 Schools

ACTIVITY — HAMBURGER ICON

EXERCISE

KEY OBJECTIVE

- Practice using CSS transitions

TYPE OF EXERCISE

- Individual/Partner Lab

TIMING

10 min	1. Follow the instructions in **lesson11_starter_code > [3] transformation_lab**

ANIMATIONS

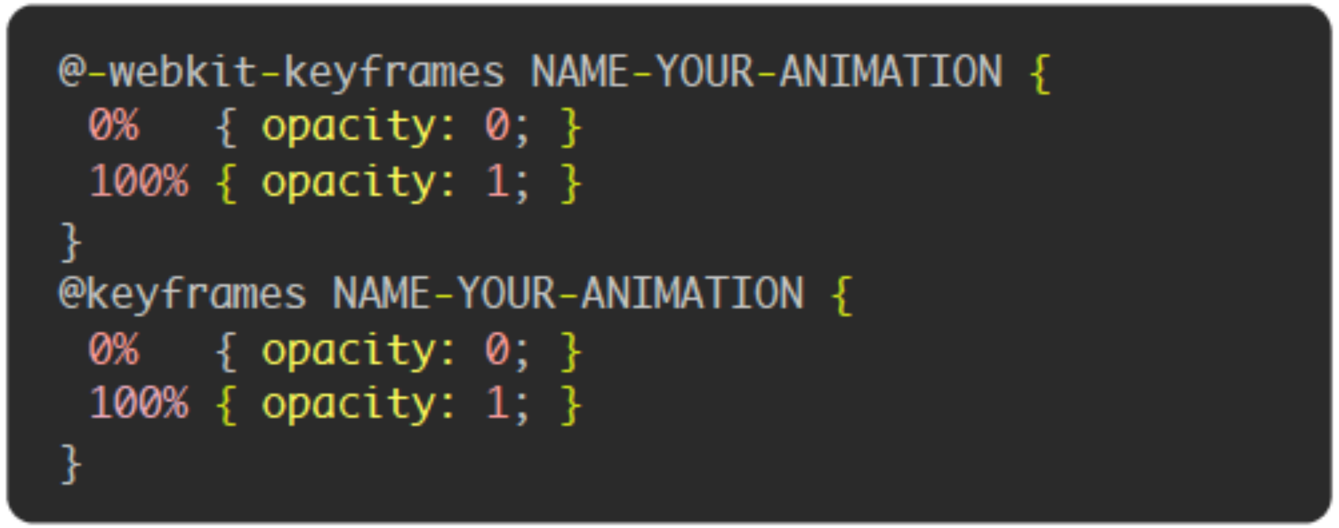
KEYFRAME ANIMATIONS

- Keyframe animations allow developers to create smooth, maintainable animations that perform well and don't require tons of scripting

https://www.impressivewebs.com/demo-files/css3-animated-scene/

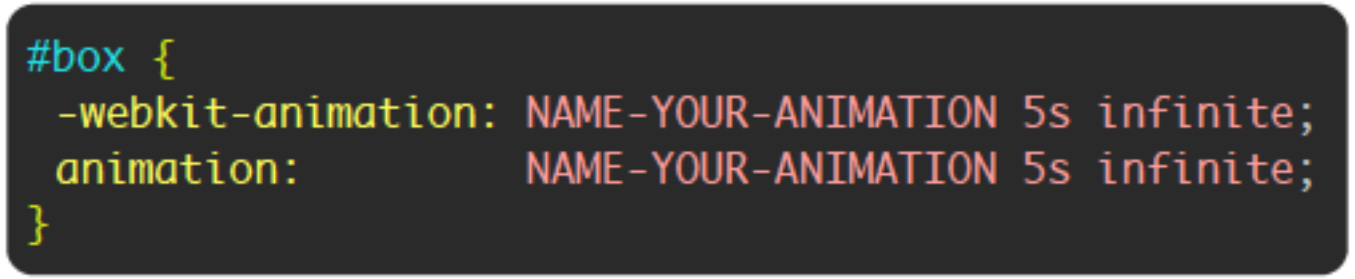
KEYFRAME ANIMATIONS — SYNTAX

1. Define custom animation



```
@-webkit-keyframes NAME-YOUR-ANIMATION {
  0%   { opacity: 0; }
  100% { opacity: 1; }
}
@keyframes NAME-YOUR-ANIMATION {
  0%   { opacity: 0; }
  100% { opacity: 1; }
}
```

2. Assign using the animation property



```
#box {
  -webkit-animation: NAME-YOUR-ANIMATION 5s infinite;
  animation:         NAME-YOUR-ANIMATION 5s infinite;
}
```

A simple tool to make sure you're including all the necessary browser prefixes: pleeease

KEYFRAME ANIMATIONS — ANIMATION PROPERTY

Properties:

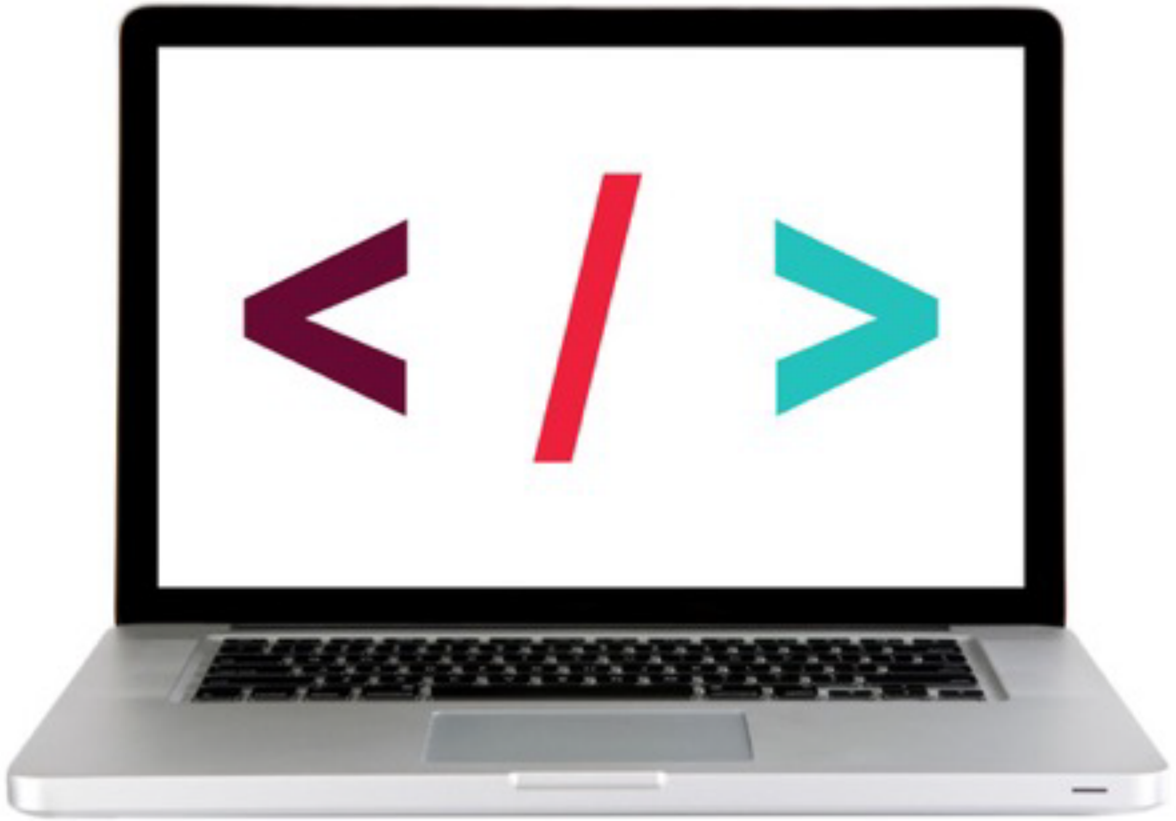
```css
.box {
 animation-name: bounce;
 animation-duration: 4s;
 animation-timing-function: ease-out;
 animation-delay: 2s;
 animation-iteration-count: 10;
}
```

Shorthand:

```css
.example {
  animation: name duration timing-function delay iteration-count;
}
```

```css
.example {
  animation: bounce 4s ease-out 2s 10;
}
```

LET'S TAKE A CLOSER LOOK — TRIGGERING TRANSITIONS



Code along — Spinning Wheel

LAB

LAB

ACTIVITY

EXERCISE

KEY OBJECTIVE

- Practice triggering CSS transitions with jQuery

TYPE OF EXERCISE

- Individual/Partner Lab

TIMING

10 min

Until 8:50

2. Add base styles to the page **lesson11_starter_code > [5] sidebar_lab**
3. Make sidebar interactive using jQuery and CSS transitions.

HOMEWORK

HAVE CSS MOSTLY READY THIS SUNDAY

FINISH INTERACTIVE NAV AND PANELS

FEWD

EXIT TICKETS