

WEB DESIGN DECAL

LECTURE 10

More jQuery

More event listeners, and event propagation

What did we do last week?

Getting and setting content with:

`.text()`, `.html()`, `.val()`

Manipulating your CSS with:

`.css()`, `.addClass()`, `.removeClass()`

Listening for page events using:

`.hover()`, `.click()`, `.scroll()`

What are we doing today?

jQuery Event Listeners

DOM traversal

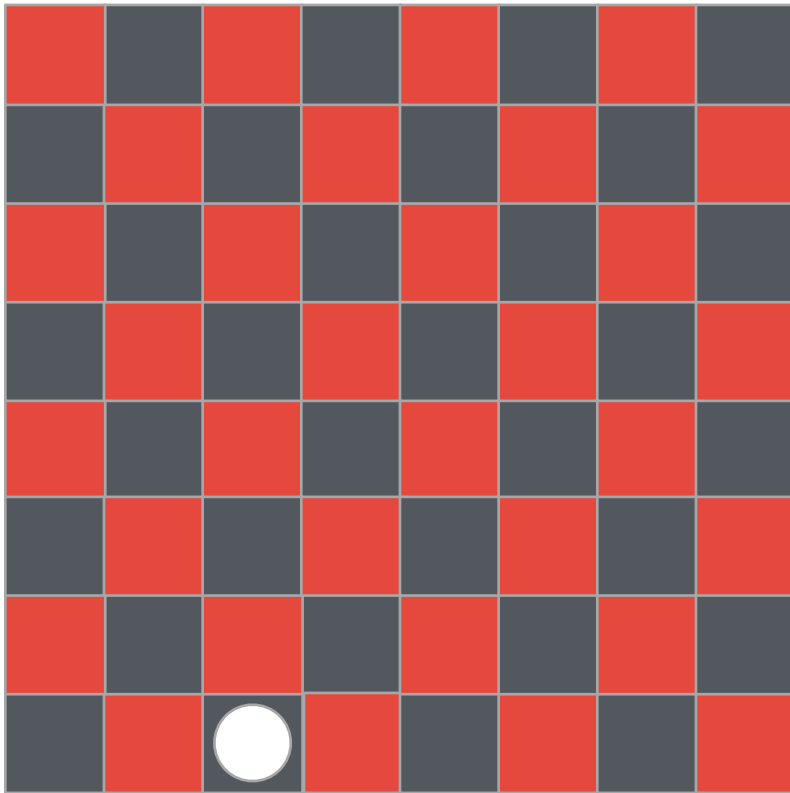
Event Objects

jQuery - A head-scratcher:

Let's say I'm making a game of checkers with HTML, CSS, and jQuery. All of the squares on my board have class `.square`.

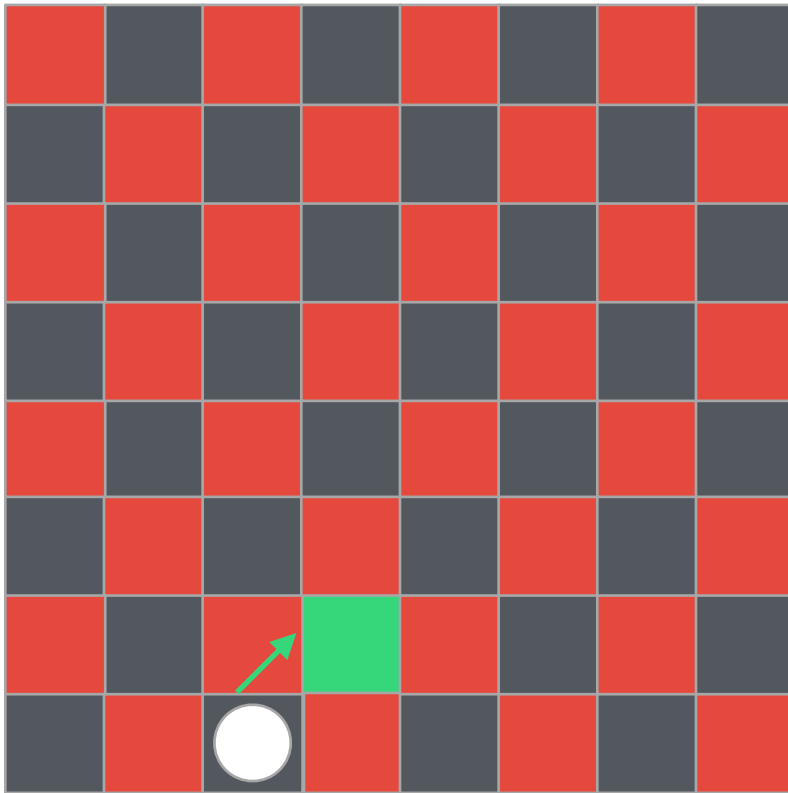
I want my player to select the square she wants to move to, and have that square light up green (indicating that's the square she chose)

jQuery - A head-scratcher:



 = Checker piece

jQuery - A head-scratcher:



○ = Checker piece

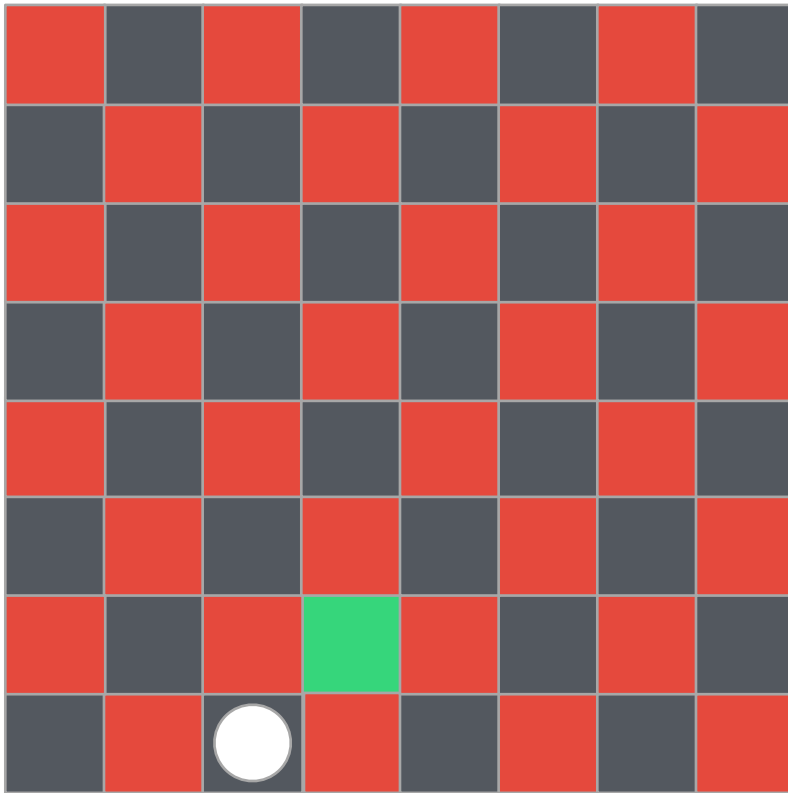
jQuery - A head-scratcher:

Here's my code:

```
$( ".square" ).click(function () {  
    if ( !$( ".square" ).hasClass( "selected" ) ) {  
        $( ".square" ).addClass( "selected" );  
    }  
});
```

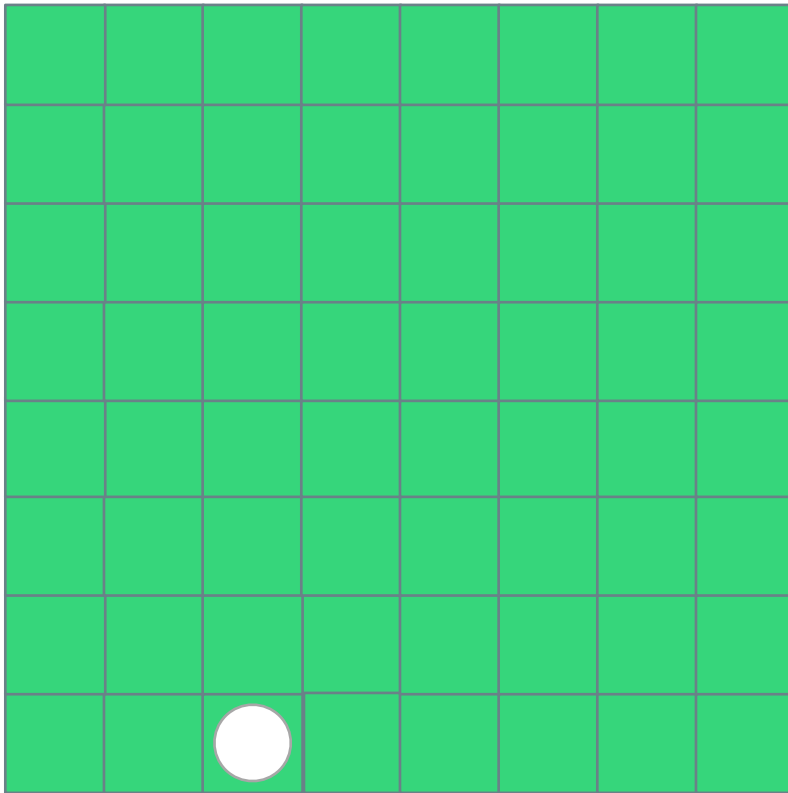
So will it work?

jQuery - A head-scratcher:



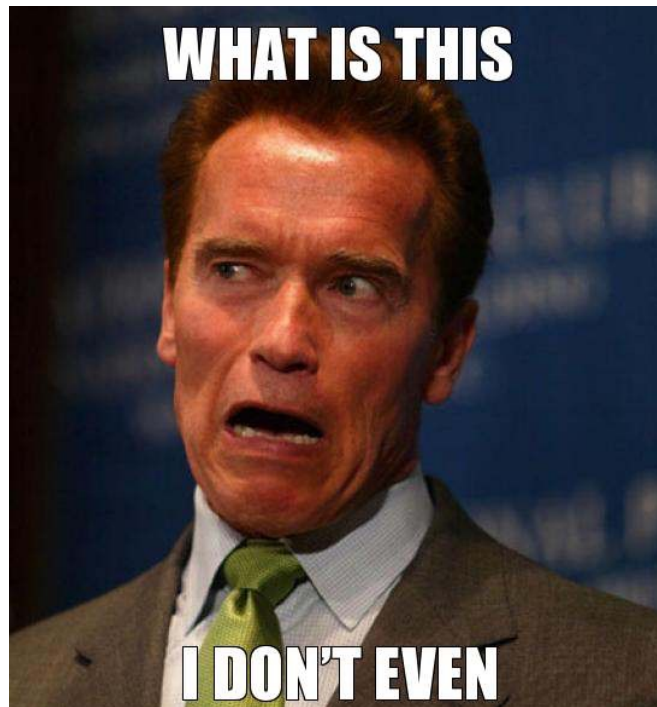
 = Checker piece

jQuery - A head-scratcher:



○ = Checker piece

jQuery - A head-scratcher:



jQuery - this

You need some way to select the particular instance of the class. This is where **this** comes in. Try this:

```
$( ".square" ).click(function () {  
    if ( !$(this).hasClass("selected") ) {  
        $(this).addClass("selected");  
    }  
});
```

Basically says, when any square is clicked, I want the one I clicked to have the “selected” class added to it.

jQuery - this

Note that this can only be used in a function body, so we know what we're talking about. This wouldn't work:

```
$(this).click(function () {  
    if (!$(this).hasClass("selected")) {  
        $(this).addClass("selected");  
    }  
}); // wat is this
```

More jQuery Event listeners

jQuery - `scrollTop()`

Say I want to make a site where things happen as you scroll down the page. I'm reading the jQuery documentation, but I'm not sure how things work.

Last week we briefly went over `.scroll()`:

jQuery - scrollTop()

After Googling, I learn about `.scrollTop()`. Can you tell me what it does?

```
$(window).scroll(function () {  
    var height = $(window).scrollTop();  
  
    if (height == 1000) { // once we reach 1000px mark  
        alert("We've made it to 1000");  
        // or do whatever you want here  
    }  
});
```

jQuery - `scrollTop()`

So now we've discovered that `scrollTop` tells us how far we are from the top of the browser.

How would you make a “sticky” navigation bar like this?

How can we check to see if we're right?

jQuery - scrollTop()

So this can get the height from the top of the page, but can we also set the height from the top of the page?

If so, how do we do it?

```
$("#about-me").click(function () {  
    var aboutMe = 1200;  
  
    $(window).scrollTop(aboutMe);  
})
```

jQuery - scrollTop()

So now we've discovered that `.scrollTop()` tells us how far we are from the top of the browser.

What if we wanted to see how far we are from the left side of the browser?

jQuery - offset() & position()

Using the code already written in this jsFiddle, tell me what each of these two jQuery properties do.

tinyurl.com/OffsetAndPosition

jQuery - offset() & position()

This way we can now get the x and y coordinates of any element on the page, relative to its parent or absolute.

Can you also set these just like you can get these?

jQuery - offset() & position()

Of course you can, just try adding values for top and left:

```
$( "#elem" ).offset( { top: 50, left: 100 } );
```

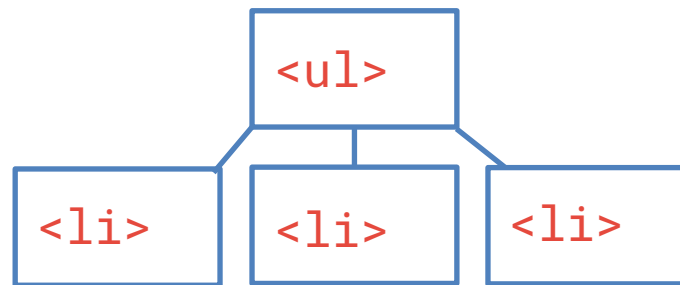
This is all super helpful for animations.

DOM Traversal

```
1  <html>
2    <head>
3      <title></title>
4    </head>
5    <body>
6      <div></div>
7      <div>
8        <ul>
9          <li></li>
10         <li></li>
11         <li></li>
12       </ul>
13     </div>
14     <div></div>
15   </body>
16 </html>
```


jQuery - parent ()

The element in the level above the element you've selected

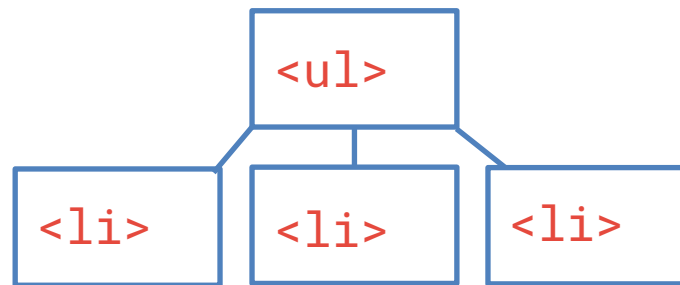


In this example, the parent is `` if you're querying any of the ``'s.

`$("li").parent()` -> ``

jQuery - siblings()

Returns a list of the elements on the same level as you, sharing the same parent.

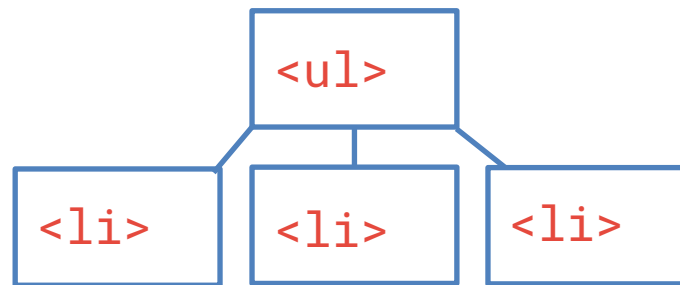


In this example, the ``'s are all siblings of one another.

```
$("li").siblings() -> <li>, <li>
```

jQuery - children()

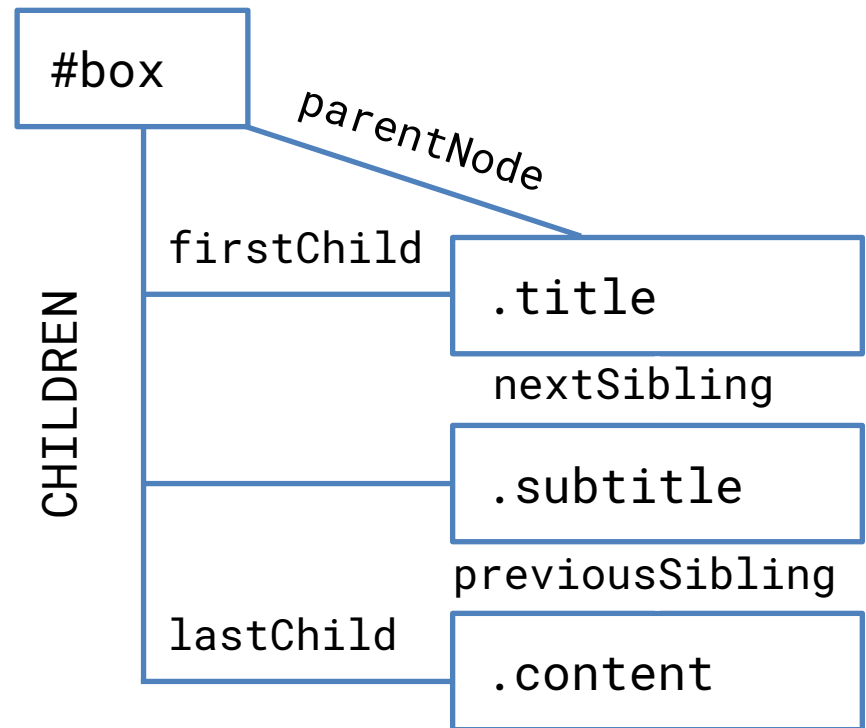
Returns a list of the elements on the level directly below yours.



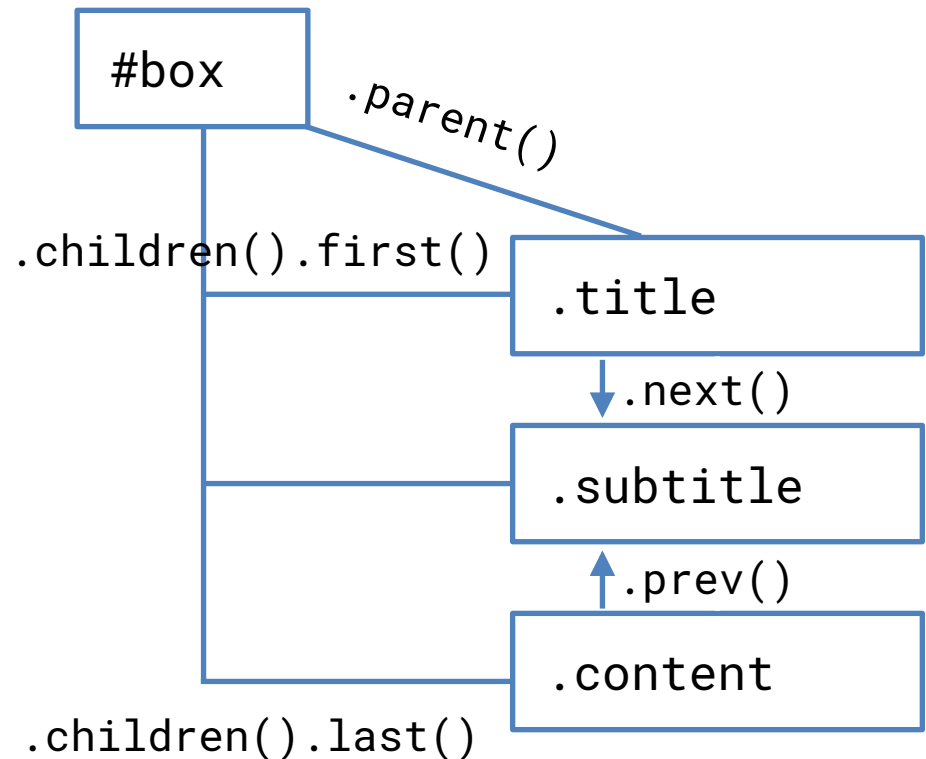
In this example, querying the `` would return a list of all the ``'s.

```
$( "ul" ).children() -> <li>, <li>, <li>
```

```
<div id="box">
  <div class="title">
    Hamazing
  </div>
  <div class="subtitle">
    Hamza is the bomb-za
  </div>
  <div class="content">
    Play that sax, Phil
  </div>
</div>
```



```
<div id="box">
  <div class="title">
    Hamazing
  </div>
  <div class="subtitle">
    Hamza is the bomb-za
  </div>
  <div class="content">
    Play that sax, Phil
  </div>
</div>
```



```
<ul id="menu">
  <li>Tacos</li>
  <li>Pad Thai</li>
  <li>Lasagna</li>
  <li>Pizza</li>
</ul>
```

Given `$("#menu")`, how would you get _____?

jQuery - method chaining

Don't forget that it's perfectly valid for there to be “chains” of methods.

For instance:

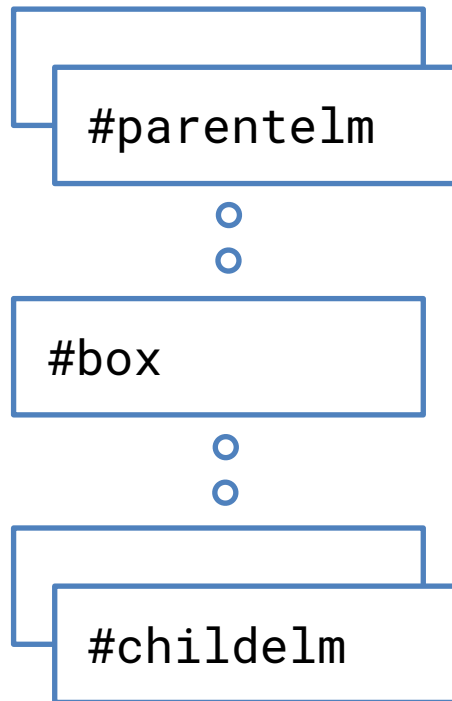
```
$( "#menu" ).children().first().parent();
```

which is exactly the same as:

```
$( "#menu" )
```

jQuery - why DOM traversal matters

Rearranging code doesn't have to break everything!

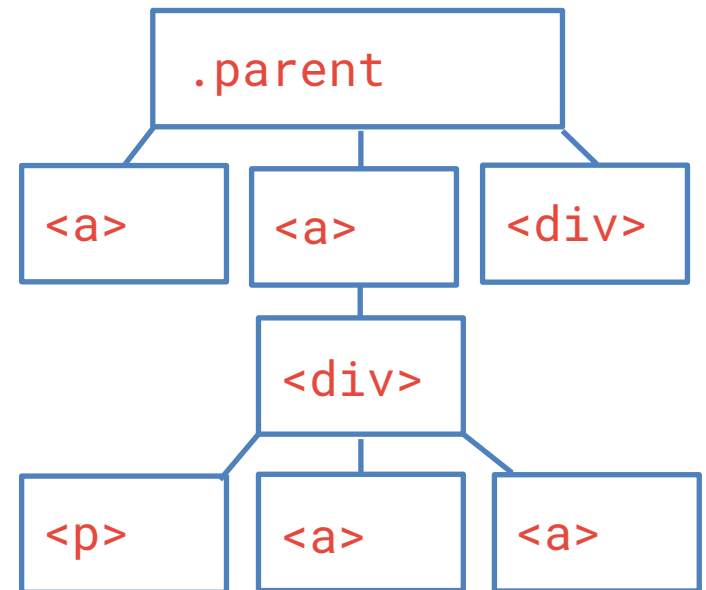


jQuery - find()

Search through the children of the element to find something else

```
$( ".parent" ).find( "a" );
```

Returns all of the `<a>` elements inside `.parent`, regardless of how “deep” it’s nested

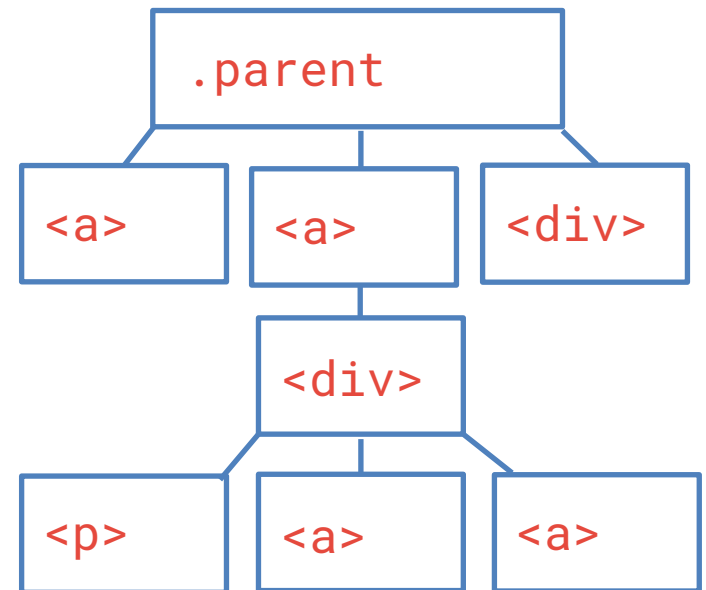


jQuery - find()

One caveat:

```
$( ".parent" ).find( "a" );
```

Using `.find()` can be really slow, so use it sparingly!



Event Objects?

Let's say I've got this modal up where I want to click on the surrounding overlay to close it.



```
<div class="overlay">  
  <div class="modal">  
    This is a modal!  
  </div>  
</div>
```

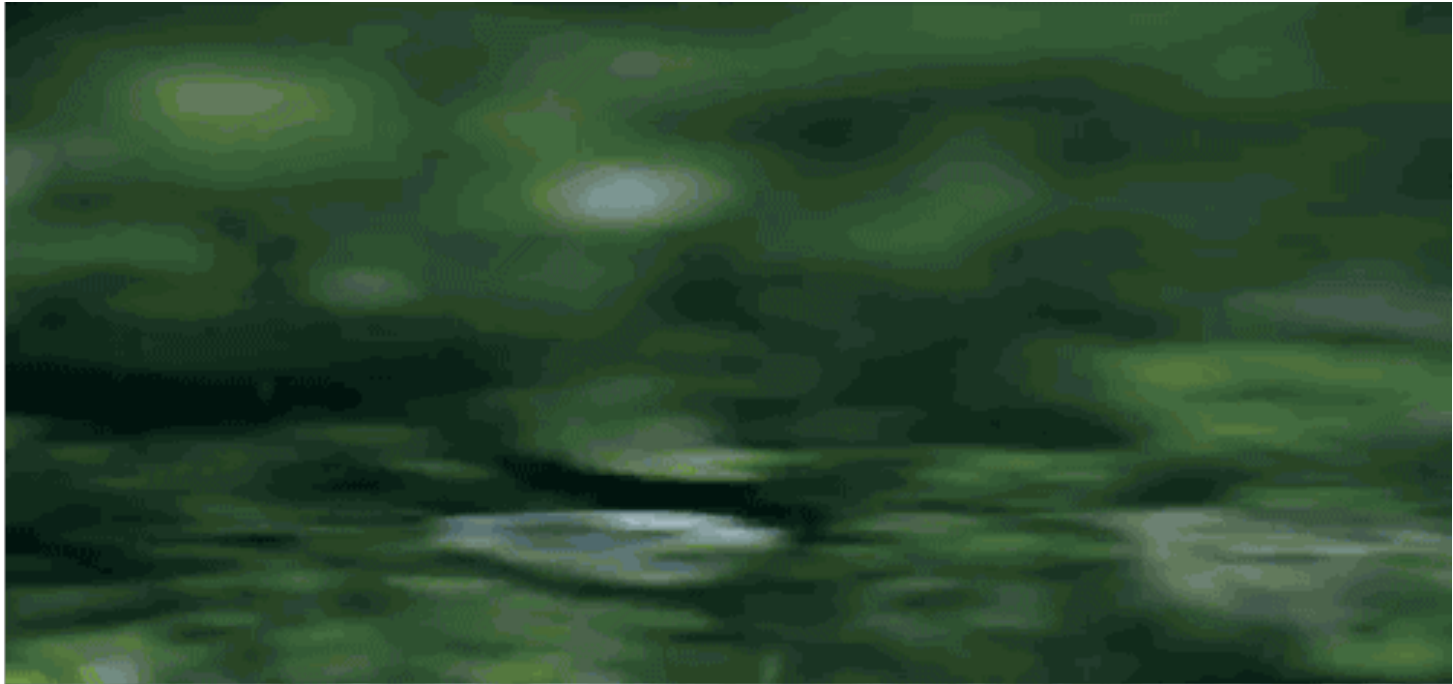
```
$(".overlay").click(  
  function () {  
    $(this).fadeOut();  
  });
```



But here's my problem, even clicking on the modal will make everything disappear:

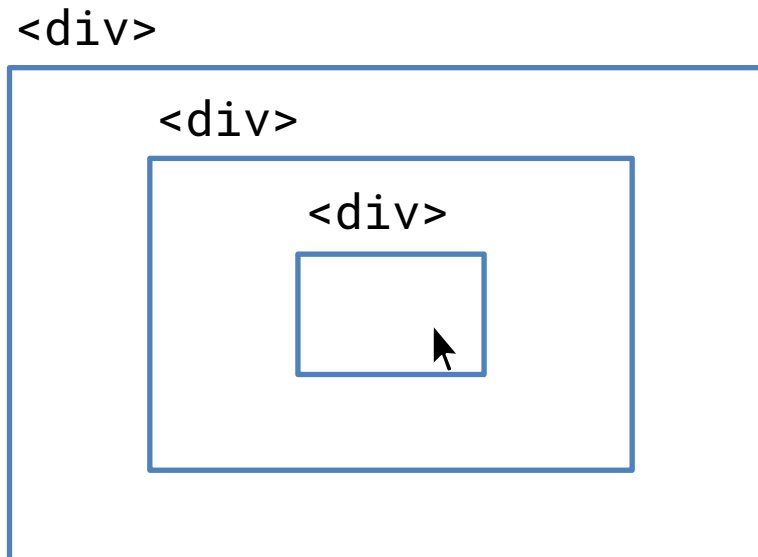


This is really a “ripple” effect of programmed events



Event Propagation

When you click on an element, the click will “bubble” to each of its parents. Each parent will know that a click occurred.



jQuery - stopPropagation()

Use `.stopPropagation()` to prevent the event from bubbling outward.

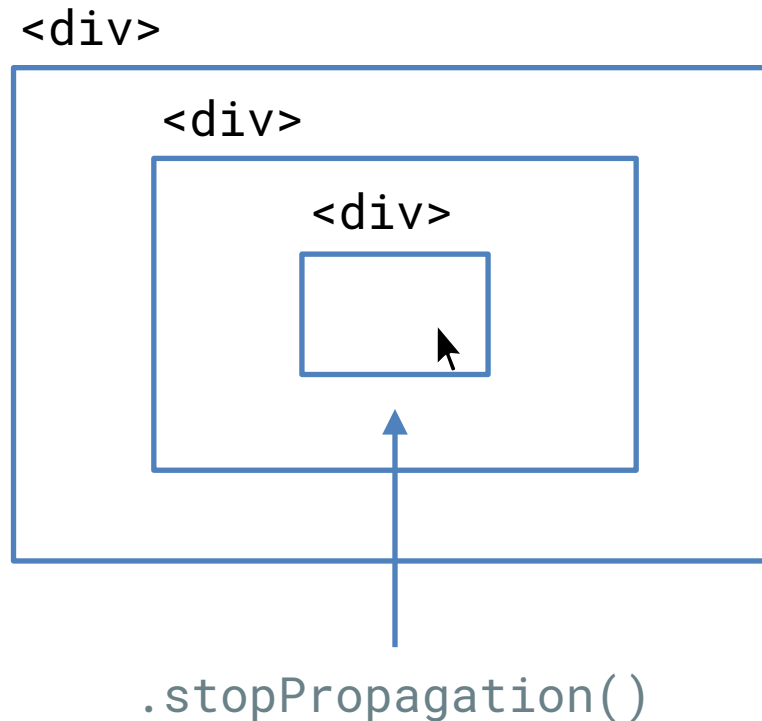
```
<div class="overlay">
  <div class="modal">
    This is a modal!
  </div>
</div>
```

```
$(".overlay").click(
  function () {
    $(this).fadeOut();
  })

$(".modal").click(
  function (event) {
    event.stopPropagation();
  })
```

jQuery - stopPropagation()

Use `.stopPropagation()` to prevent the event from bubbling outward.



jQuery - preventDefault()

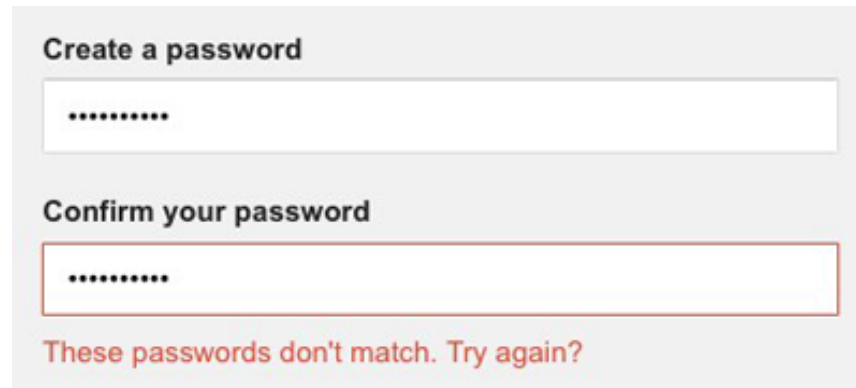
.preventDefault() will entirely stop the default behavior.

```
<a id="link" href="http://google.com">  
  Don't take me to Google!  
</a>
```

```
$("#link").click(  
  function (event) {  
    event.preventDefault();  
  });
```

jQuery - stopPropagation()

Really useful for when you want to stop events on a page.
Like when you want to stop a form submission (if the passwords don't match, for example).



Create a password

.....

Confirm your password

.....

These passwords don't match. Try again?

Any questions?