

# jQuery – JavaScript library

Samkeun Kim <skim@hknu.ac.kr>

<http://cyber.hknu.ac.kr/>

# Head First jQuery

Quickly  
implement  
complex  
HTML forms



Build web apps  
without pesky  
browser plug-ins



Write your  
own custom  
jQuery  
functions



Add interactivity to  
your web pages in just  
a few lines of code



Put animation  
and Ajax to use in  
your web pages



# Getting Started with jQuery: Web Page Action

# You want web page power

You already know how to build great-looking web pages with clean, valid HTML and CSS. But **static web pages just don't cut it anymore—people want a responsive web page.** They want action, animation, interaction, and lots of cool effects.

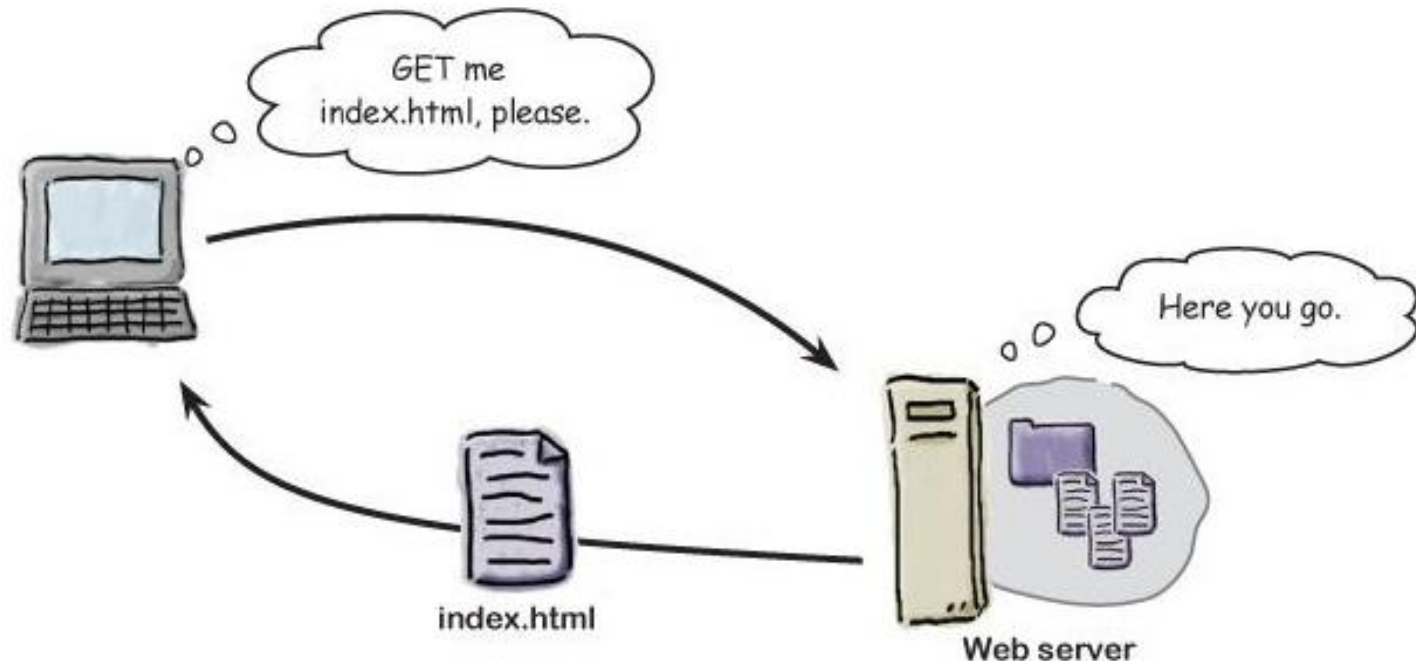


# HTML and CSS are fine, but...

Plain old HTML and CSS are good for giving your page structure and style.

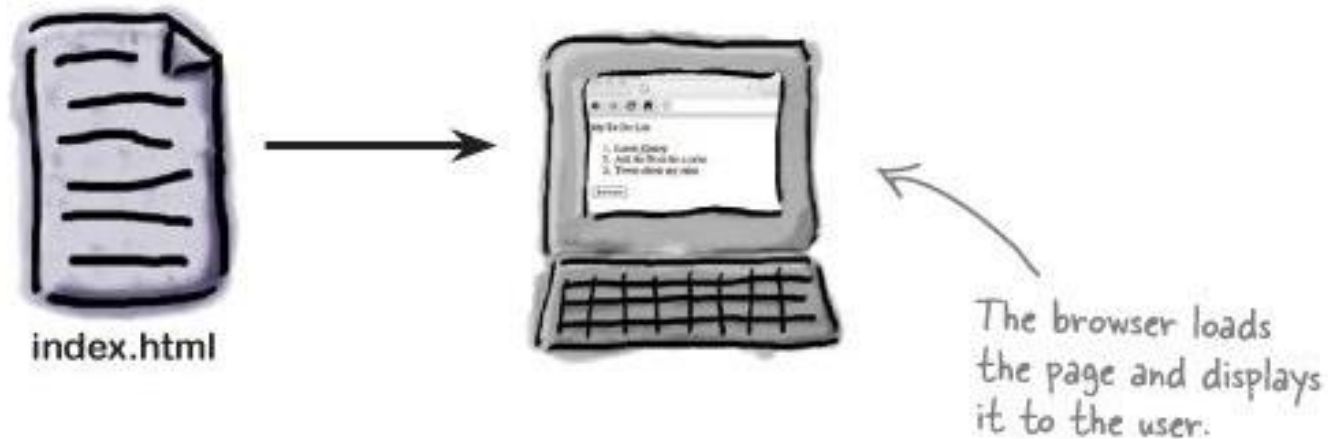
Once you have a rendered HTML page, it's there, but it's **static**.

1. **Browser requests a web page from a server** when someone types a web address into the browser's URL bar.



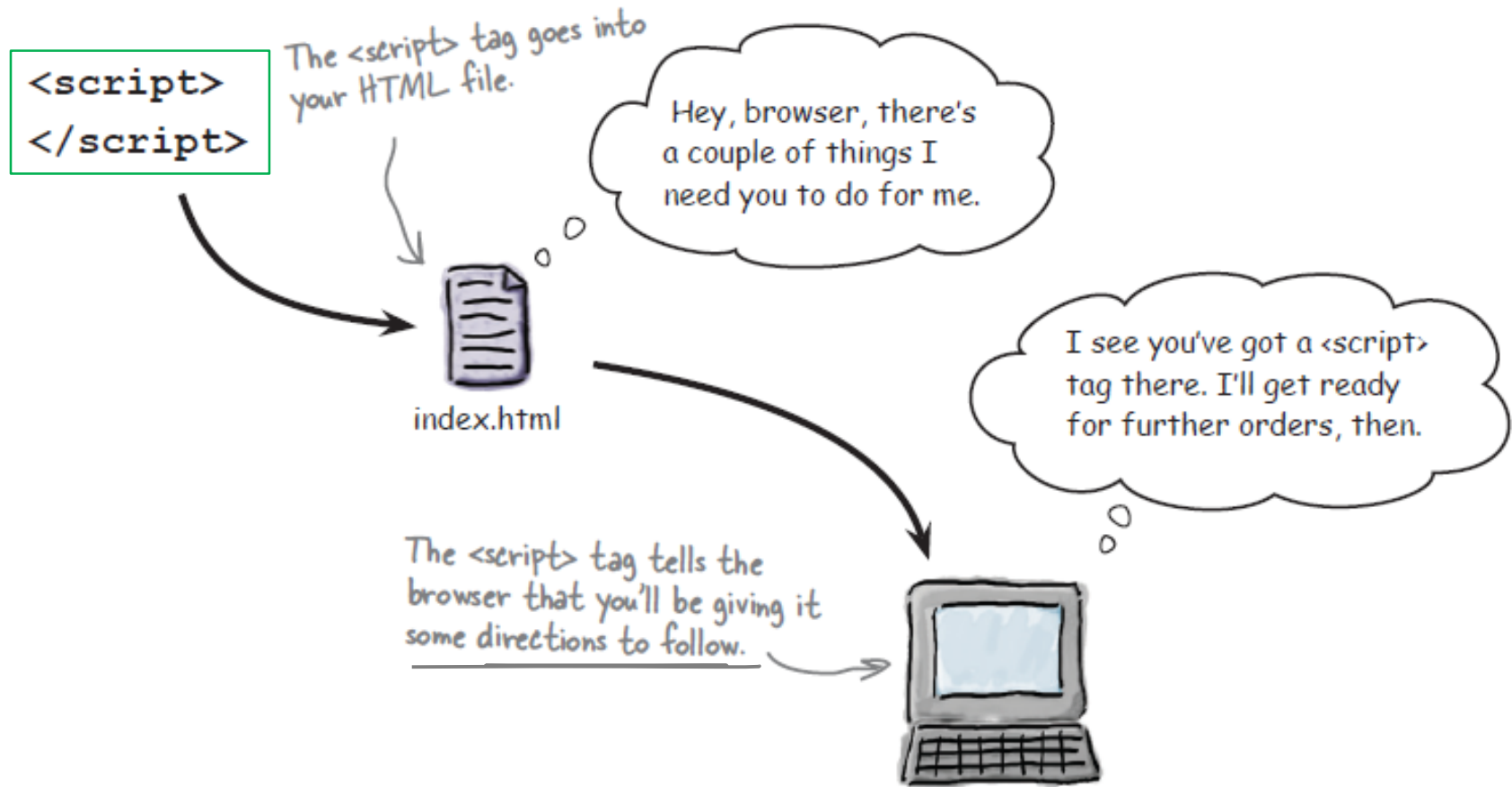
# HTML and CSS are fine, but...

2. **Server finds the requested file** and sends them to the browser
3. **Browser displays a rendered HTML page** based on the file sent from the server




# ...you need the power of script

To change your web pages on the fly, without reloading, you need to talk to your browser. How do you pull that off ? With an HTML tag known as `<script>`.



# ...you need the power of script



But how do I give the browser directions? That seems kinda unusual...

Great question. Remember that HTML is a markup language that handles document structure.

And cascading style sheets (CSS) control the look and feel and position of those elements.

**HTML and CSS** control how a web page is built and displayed, but **neither of them can add behavior to the web page.**

What we need for that is a **scripting language.**

What we need is **jQuery**.



# Enter jQuery (and JavaScript)!

Every browser comes with a **built-in JavaScript interpreter** that takes the directions you write in between the `<script>` tags and translates those directions into different kinds of action on the web page



jQuery is a JavaScript library specialized for changing web page documents on the fly.

# jQuery's CDN or Download

jQuery's CDN or Download: <https://code.jquery.com/>

jQuery 3.x

*jQuery Core 3.4.1*

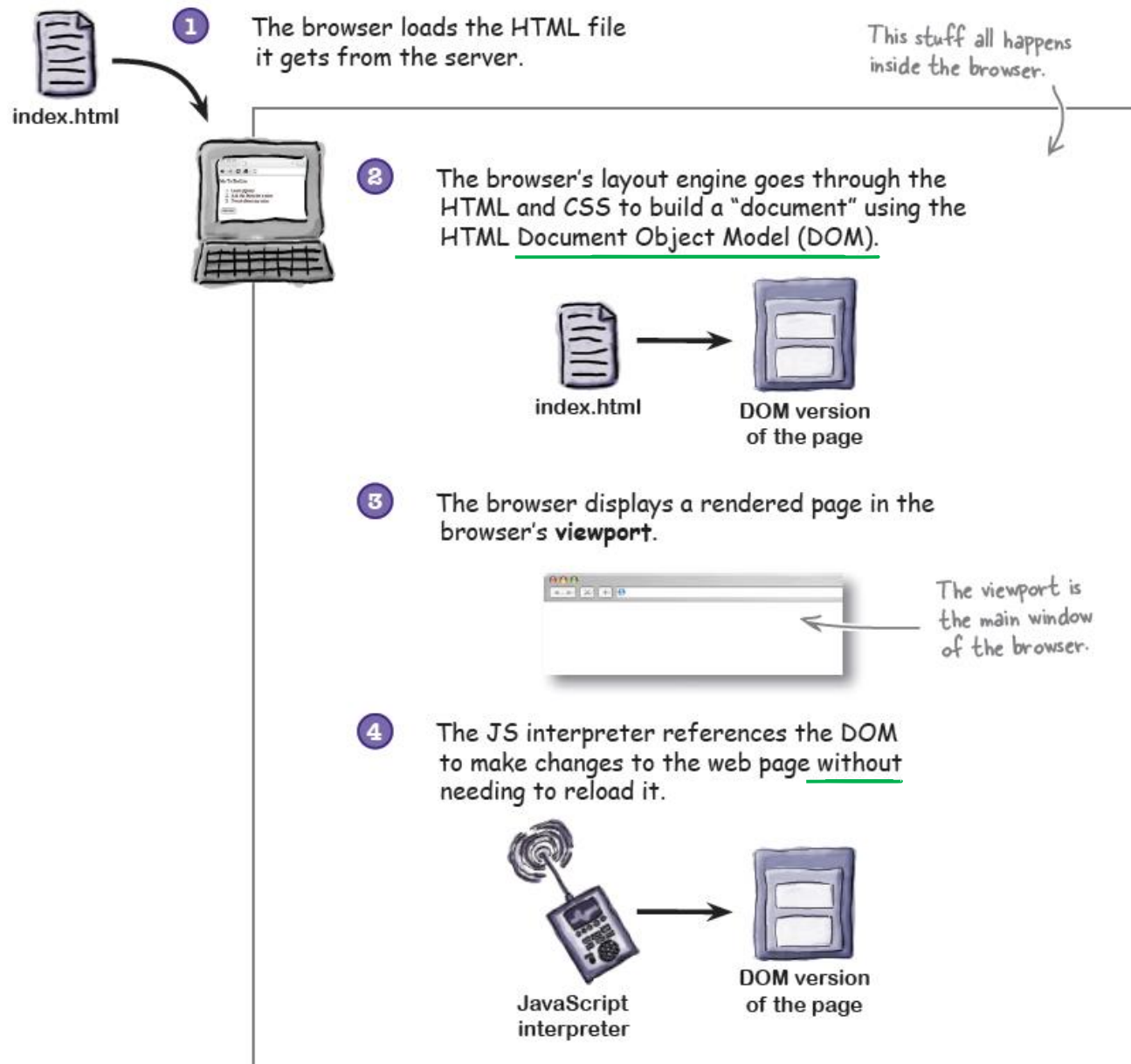
```
<script src="https://code.jquery.com/jquery-3.4.1.js"></script>
```

**Downloading jQuery:**

<http://jquery.com/download/>

예: 적절한 디렉토리에 저장 => `<script src="저장된 디렉토리/jquery-3.4.1.min.js"></script>`

# Look into the browser





# jQuery makes the DOM less scary

jQuery keeps DOM simple.

Don't forget: jQuery is JavaScript, but a **much more approachable version**.

## The raw JavaScript way

I'm talking to the document (aka the big D in DOM).

Get me all of the elements that have the tag name of "p."

```
document.getElementsByTagName("p")  
[0].innerHTML = "Change the page.";
```

Get me the zeroth element.

Set the HTML inside that element... ..to this stuff.

## The jQuery way

Grab me a paragraph element.

Change the HTML of that element to what's in these parentheses.

```
$("#p").html("Change the page.");
```

jQuery uses a "selector engine," which means you can get at stuff with selectors just like CSS does.

Let's say we want to change the HTML inside of **five paragraph elements** on our page:

Loop through the number of elements I want to change.

```
for (i = 0; i <= 4; i++)  
{  
    document.getElementsByTagName("p")  
[i].innerHTML="Change the page";  
}
```

Get me the element we're looping over.

Because jQuery uses CSS selectors, we can say it the same way as above.

```
$("p").html("Change the page.");
```

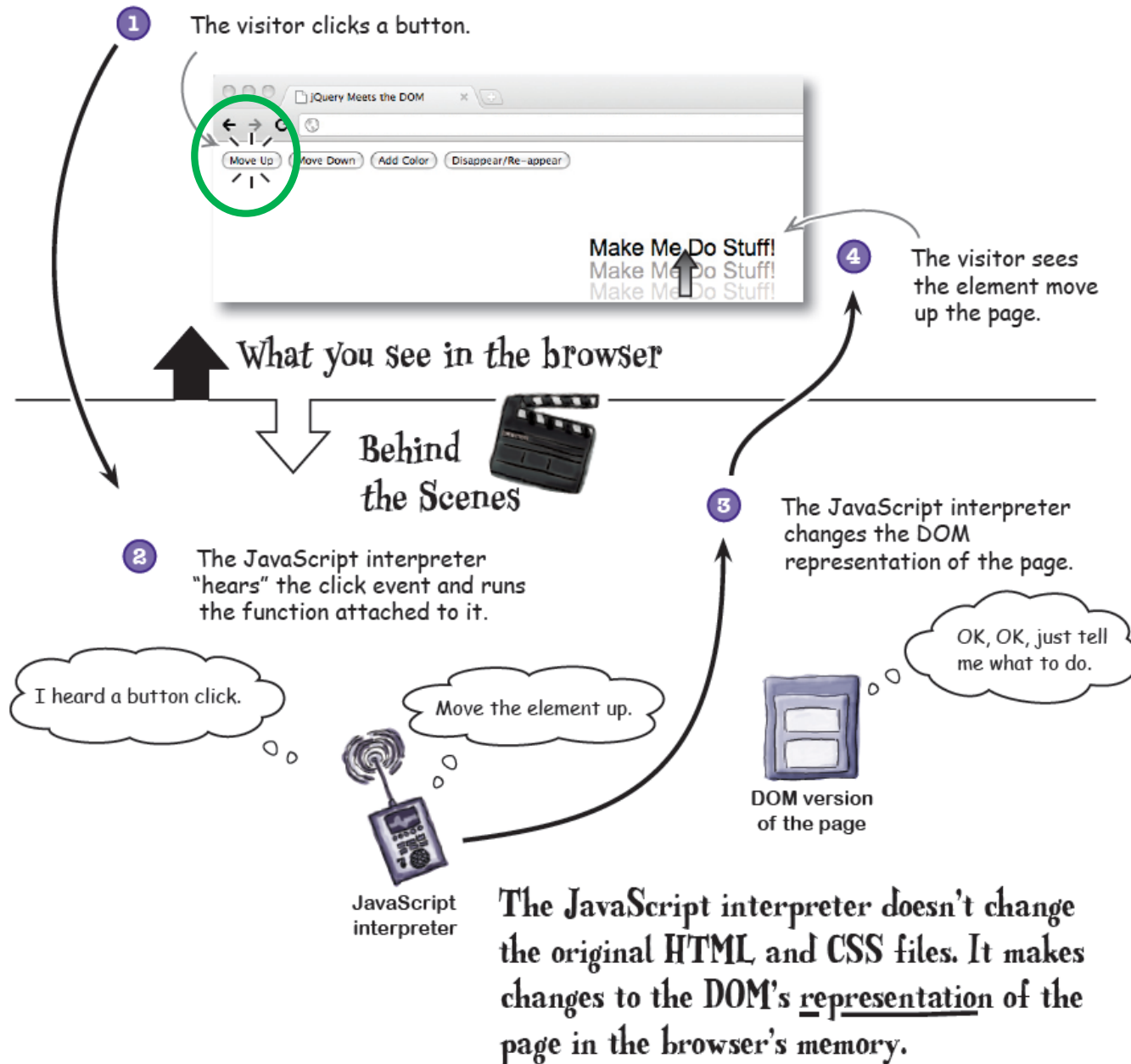
One of jQuery's main strengths is that it **allows you to work with the DOM without having to know every little thing about it**. Underneath it all, JavaScript is doing the heavy lifting.

# READY BAKE CODE

*jQ\_meets\_DOM.html*

```
1  <!DOCTYPE html>
2  <html><head> <title>jQuery Meets the DOM</title>
3  <style>
4  #change_me {position: absolute; top: 100px; left: 400px; font: 24px arial;}
5  #move_up #move_down #color #disappear {padding: 5px;}
6  </style>
7  <script src=scripts/jquery-1.5.2.min.js></script>
8  </head>
9  <body>
10     <button id="move_up">Move Up</button>
11     <button id="move_down">Move Down</button>
12     <button id="color">Add Color</button>
13     <button id="disappear">Disappear/Re-appear</button>
14
15     <div id="change_me">Make Me Do Stuff!</div>
16 </script>
17 $(document).ready(function() {
18     $("#move_up").click( function() {
19         $("#change_me").animate({top:30},200);
20     });//end move_up
21     $("#move_down").click( function() {
22         $("#change_me").animate({top:500},2000);
23     });//end move_down
24     $("#color").click( function() {
25         $("#change_me").css("color", "purple");
26     });//end color
27     $("#disappear").click( function() {
28         $("#change_me").toggle('slow');
29     });//end disappear
30 });//end doc ready
31 </script>
32 </body>
33 </html>
```

# How does that work?

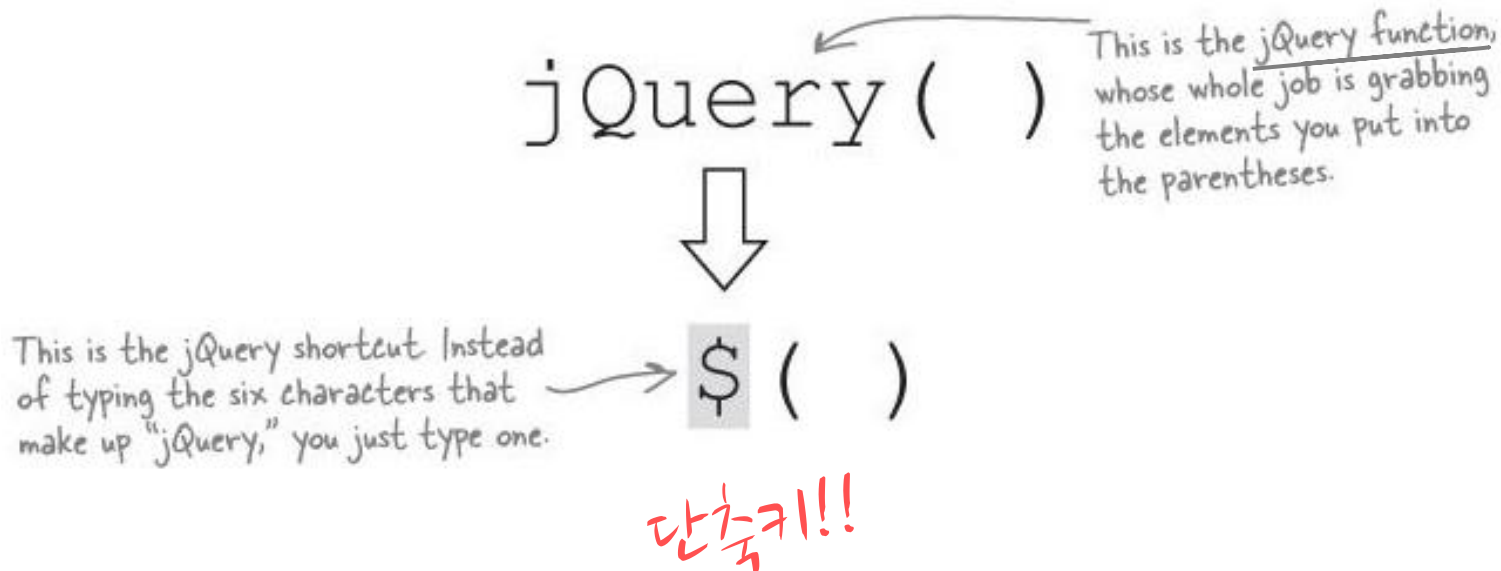




# Introducing the jQuery function

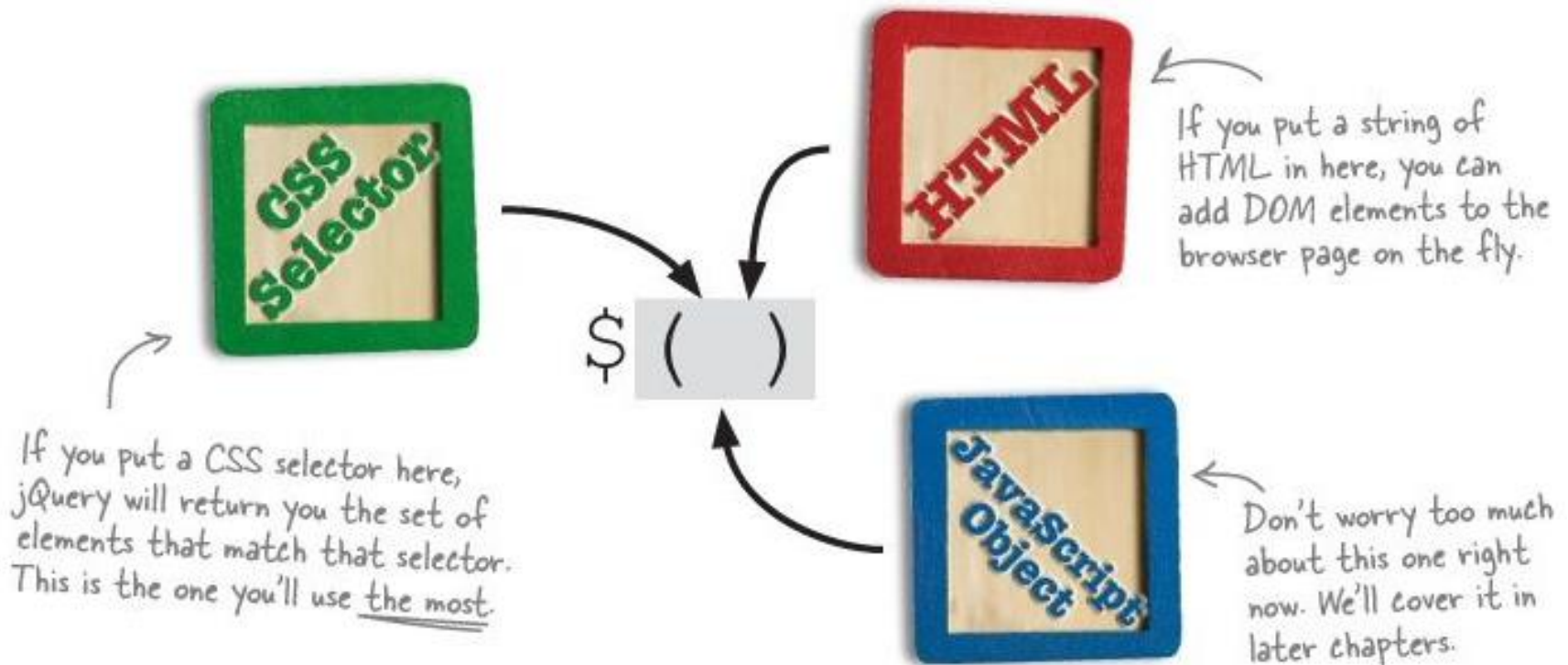
The dollar sign with the parentheses is the shorter name of the jQuery function.

This shortcut saves us from writing "**jQuery()**" *every time we want to call the jQuery function*. jQuery function is also often referred to as the jQuery wrapper.

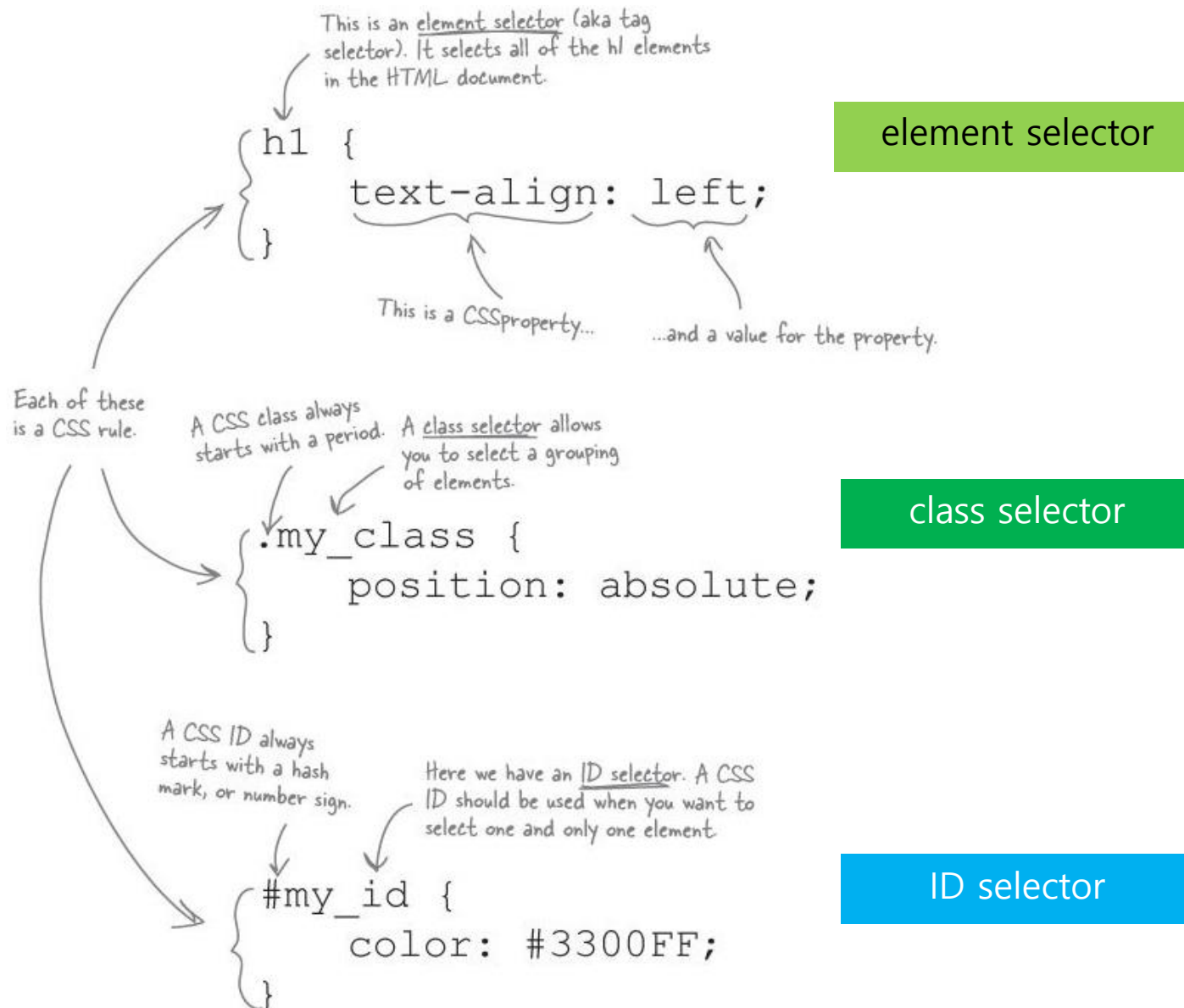


The short name and the long name point to the same thing:

the big code block known as jQuery. Throughout this book, we'll use the shortcut. Here are **three different things** you can put into the jQuery function.



# jQuery selects elements the same way **CSS** does



# Style, meet script

## CSS selector

Element selector

↓  
`h1 {  
 text-align: left;  
}`

Class selector

↓  
`.my_class {  
 position: absolute;  
}`

ID selector

↓  
`#my_id {  
 color: #3300FF;  
};`

## jQuery selector

jQuery element selector

↓      Method  
`$("h1").hide();`  
↑  
This hides all of the h1 elements on the page.

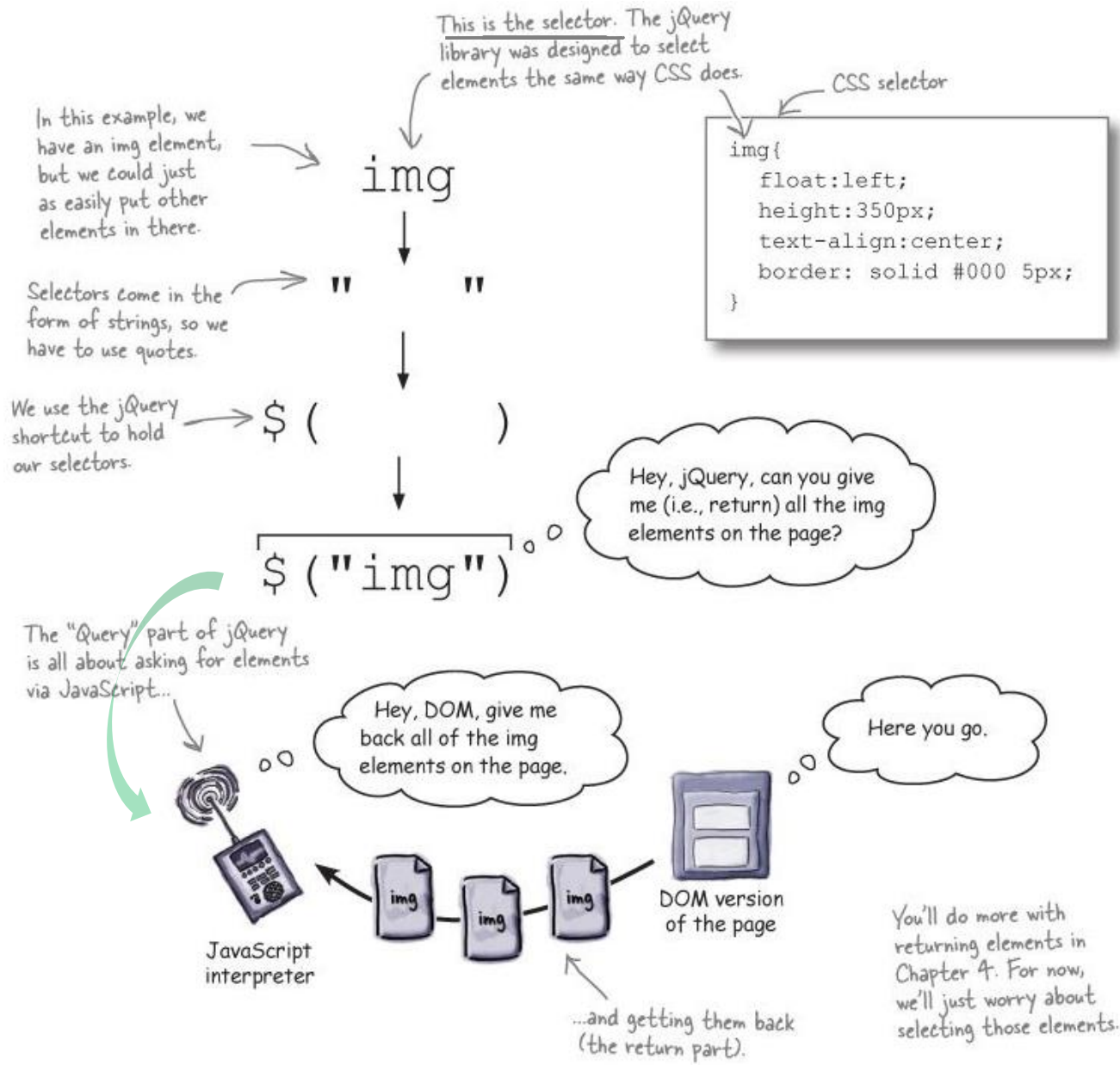
jQuery class selector

↓      Method  
`$(".my_class").slideUp();`  
↑  
Slides up all of the elements that are members of the CSS class my\_class

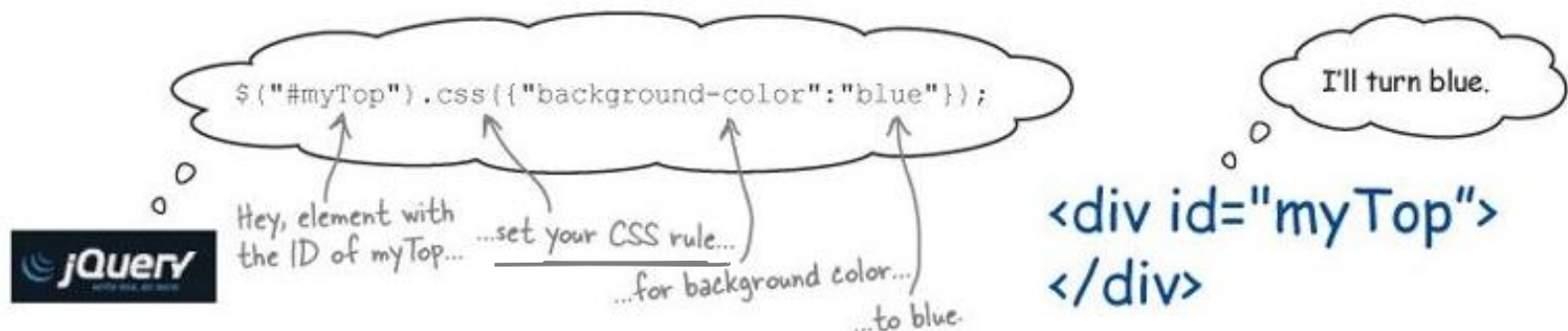
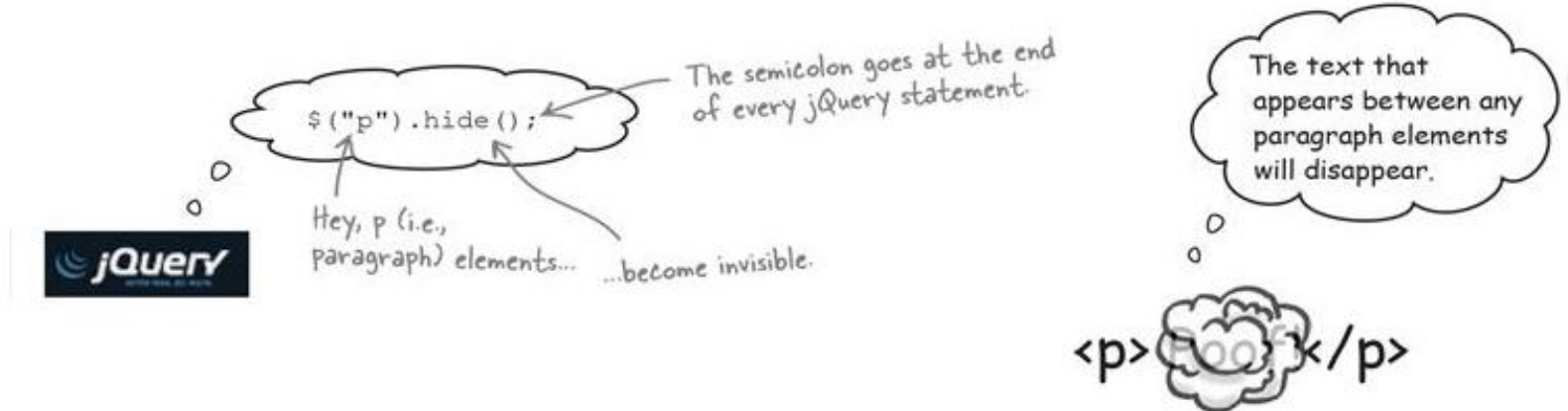
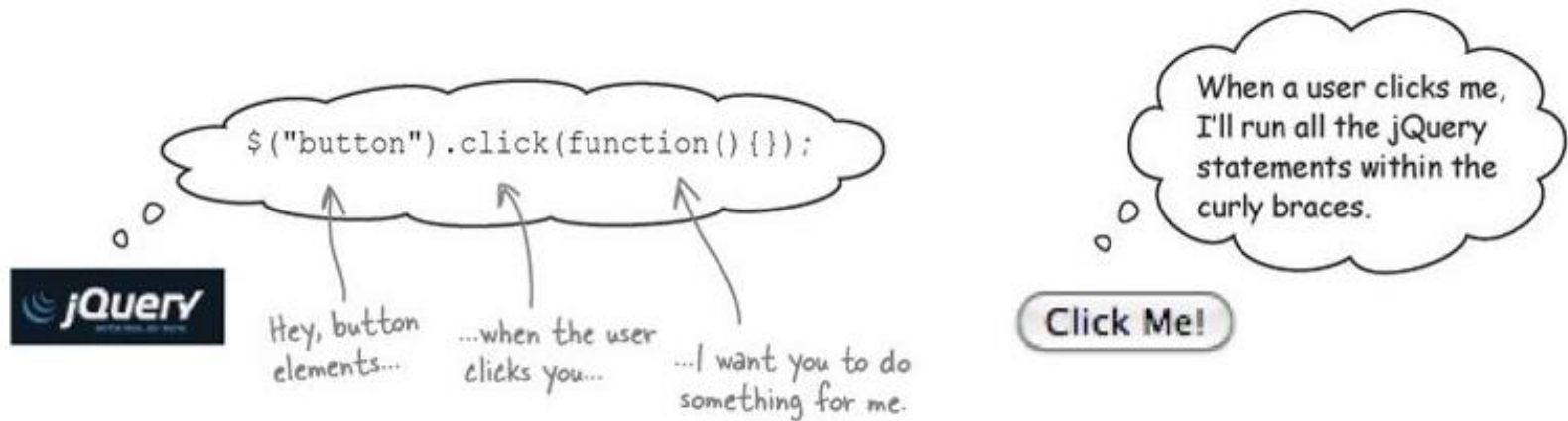
jQuery ID selector

↓      Method  
`$("#my_id").fadeOut();`  
↑  
And this jQuery statement fades out an element that has a CSS ID of my\_id until it's invisible.

# jQuery selectors at your service



# jQuery in translation



# jQuery Selectors

## element Selector

- The jQuery element selector selects elements **based on the element name**.
- You can select all <p> elements on a page like this:

```
$("#p")
```

```
$(document).ready(function(){  
    $("#button").click(function(){  
        $("#p").hide();  
    });  
});
```

[Try it yourself >>](#)

# #id Selector

- jQuery #id selector **uses the id attribute of an HTML tag** to find the specific element
- id should be unique within a page, so you should use the #id selector when you want to find a single, unique element

```
$("#test")
```

```
$(document).ready(function(){  
    $("button").click(function(){  
        $("#test").hide();  
    });  
});
```

[Try it yourself >>](#)



# .class Selector

- **jQuery class selector finds elements with a specific class**
- To find elements with a specific class, write a period character, followed by the name of the class:

```
$(".test")
```

```
$(document).ready(function(){  
    $("button").click(function(){  
        $(".test").hide();  
    });  
});
```

[Try it yourself >>](#)

## More Examples of jQuery Selectors

Syntax	Description	Example
<code>\$("*")</code>	Selects all elements	<a href="#">Try it</a>
<code>\$(this)</code>	Selects the current HTML element	<a href="#">Try it</a>
<code>\$("p.intro")</code>	Selects all <code>&lt;p&gt;</code> elements with <code>class="intro"</code>	<a href="#">Try it</a>
<code>\$("p:first")</code>	Selects the first <code>&lt;p&gt;</code> element	<a href="#">Try it</a>
<code>\$("ul li:first")</code>	Selects the first <code>&lt;li&gt;</code> element of the first <code>&lt;ul&gt;</code>	<a href="#">Try it</a>
<code>\$("ul li:first-child")</code>	Selects the first <code>&lt;li&gt;</code> element of every <code>&lt;ul&gt;</code>	<a href="#">Try it</a>
<code>\$("[href]")</code>	Selects all elements with an <code>href</code> attribute	<a href="#">Try it</a>
<code>\$("a[target='_blank']")</code>	Selects all <code>&lt;a&gt;</code> elements with a <code>target</code> attribute value equal to <code>"_blank"</code>	<a href="#">Try it</a>
<code>\$("a[target!='_blank']")</code>	Selects all <code>&lt;a&gt;</code> elements with a <code>target</code> attribute value NOT equal to <code>"_blank"</code>	<a href="#">Try it</a>
<code>\$(":button")</code>	Selects all <code>&lt;button&gt;</code> elements and <code>&lt;input&gt;</code> elements of <code>type="button"</code>	<a href="#">Try it</a>
<code>\$("tr:even")</code>	Selects all even <code>&lt;tr&gt;</code> elements	<a href="#">Try it</a>
<code>\$("tr:odd")</code>	Selects all odd <code>&lt;tr&gt;</code> elements	<a href="#">Try it</a>

# 실습과제 18-1

P.15의 **jQ\_meets\_DOM.html**을 수행하시오.

# Your first jQuery gig

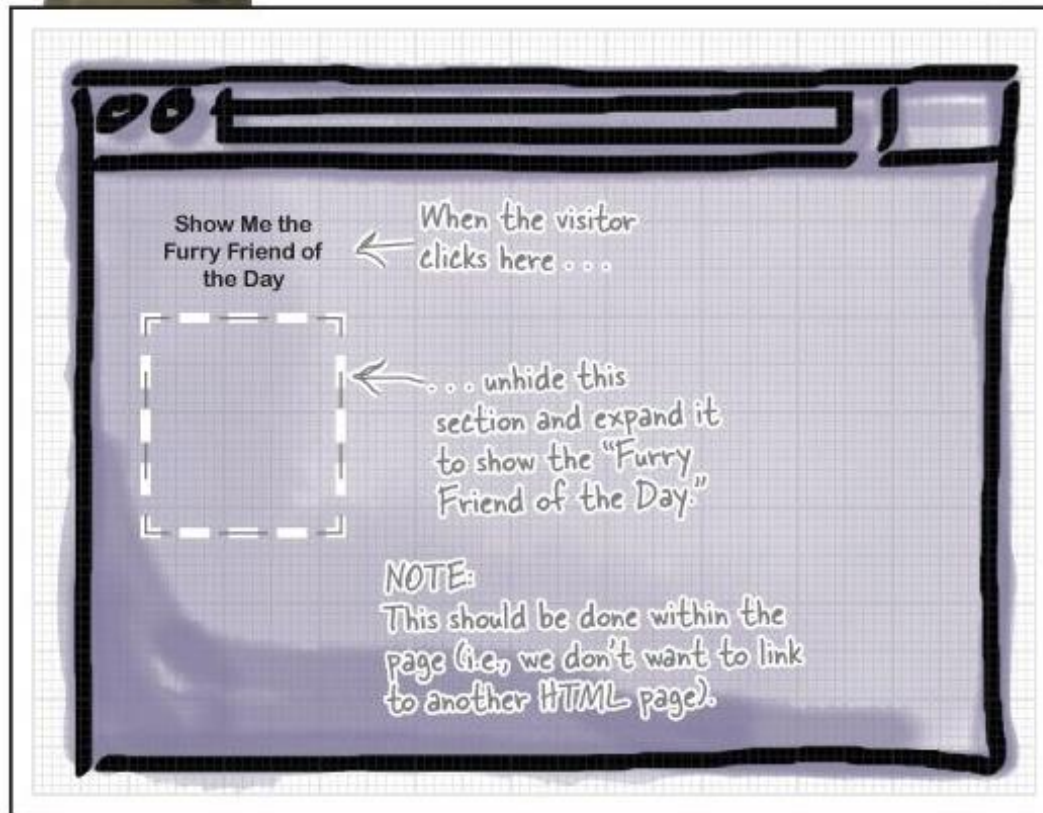


Josh from Marketing wants more interactivity.

And his manager wants richer visual effects.

We need to power up last year's page. Right now, the visitor clicks on a button and a picture pops up, but it doesn't stay on the page. We want the picture to appear when the user clicks and go away when they click again.

Also, the furry friend picture just pops up. Can you make it slide slower and sort of fade in as it does?





# Set up your HTML and CSS files

```
<!DOCTYPE html>
<html><head>
  <title>Furry Friends Campaign</title>
  <link rel="stylesheet" type="text/css" href="styles/my_style.css">
</head>
<body>
  <div id="clickMe">Show Me the Furry Friend of the Day</div>
  <div id="picframe">
    
  </div>
  <script src="https://code.jquery.com/jquery-3.4.1.js"></script>
  <script>
    $(document).ready(function() {
      $("#clickMe").click(function() {

      });
    });
  </script>
</body>
</html>
```

This makes a clickable div, and we'll style it in the CSS file below so it has the same look and feel as the picframe div.

Here's the picframe div that will slide open to show the furry friend picture

Nest the furry\_friend.jpg image inside the picframe.



index.html

```
#clickMe {  
  background: #D8B36E;  
  padding: 20px;  
  text-align: center;  
  width: 205px;  
  display: block;  
  border: 2px solid #000;  
}
```

This styles the `clickMe` div so that so it has the same look and feel as the `picframe` div.

```
#picframe {  
  background: #D8B36E;  
  padding: 20px;  
  width: 205px;  
  display: none;  
  border: 2px solid #000;  
}
```

Set the `picframe` selector to "`display: none`" so that it won't show when the page loads.



my\_style.css



As soon as I possibly can, I'll start executing code within the curly braces!



The DOM

Hey, DOM...

...whenever you're ready and loaded...

...I want you to do something for me.

```
$ (document) .ready (function () {
```

Here's our ID selector for the clickMe div.

```
$ ("#clickMe").click (function ()
```

The dot separates the selector part from the method part.

Connecting the button with an ID of clickMe to the click event, this code makes the button clickable.

The code for what will happen when the button is pressed will go between these curly braces (aka the "code block").

The semicolon is a terminator. It ends our jQuery click statement.

```
} ) ;
```

```
} ) ;
```

This semicolon ends our jQuery ready function.



You're right. Our HTML and CSS are ready; now we need some **jQuery**.

We want the **picframe div** to slide and to fade.

Fortunately, the jQuery folks have built effects that let us control both of these rich visual actions: **slides** and **fades**.

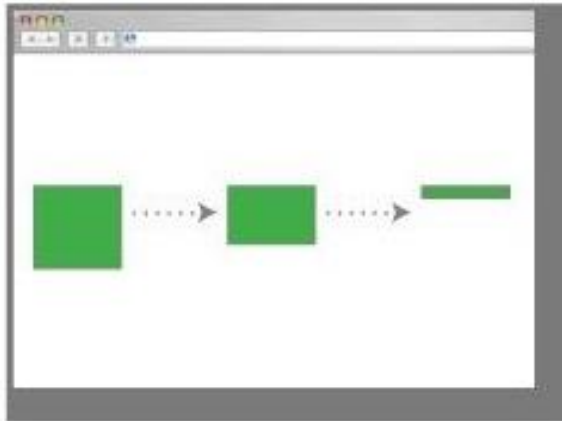
Let's just start sliding and fading first.

# Slide on in...

The first effect we'll implement is having the image slide into view, which is one of the things the marketing team manager wants to have happen.

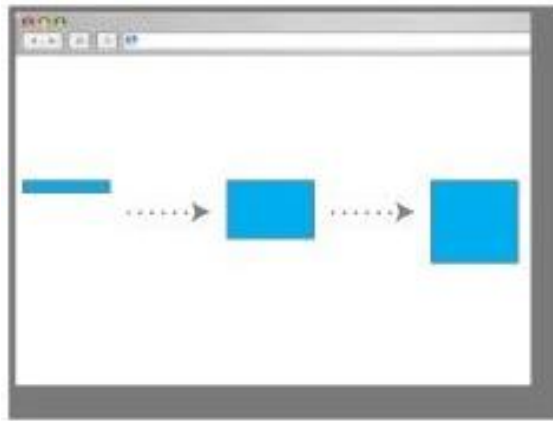
There are **three** ways to deal with **sliding**:

```
$("div").slideUp();
```



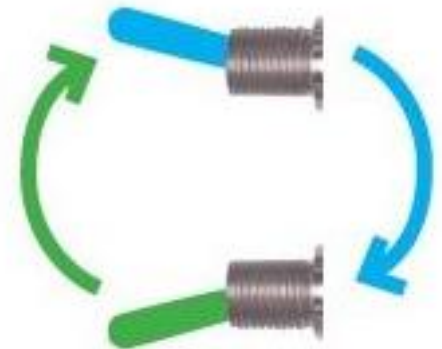
The `slideUp` method changes the height property of the element until it's 0, and then hides the element.

```
$("div").slideDown();
```



The `slideDown` method changes the height property of the element from 0 to whatever it's set to in the CSS style.

```
$("div").slideToggle();
```



The `slideToggle` action says, "If it's up, slide it down; if it's down, slide it up."

# May the fade be with you

We also want the image to gradually appear, going from invisible to fully visible. Again, jQuery has a method for that, and that method is called a **fade**.

The fade methods are pretty similar to what you just saw for sliding: you have **FadeIn**, **FadeOut**, **FadeTo**, and **FadeToggle**. For now, let's just use **FadeIn**, which gives us control over the opacity and transparency properties of HTML elements.

Here's what we want to fade in;  
in this case, it is an image.

You can specify how fast it fades in by  
putting a value inside the parentheses,  
typically represented in milliseconds (ms).

```
$("img").fadeIn();
```



When an element fades in, it goes  
from being invisible (transparent)  
to being visible (opaque).

```

<!DOCTYPE html>
<html>
  <head>
    <title>Furry Friends Campaign</title>
    <link rel="stylesheet" type="text/css" href="styles/my_style.
css">
  </head>
  <body>
    <div id="clickMe">Show me the Furry Friend of the Day</div>
    <div id="picframe">
      
    </div>
    <script src="https://code.jquery.com/jquery-3.3.1.js"></script>
    <script>
      $(document).ready(function() {
        $("#clickMe").click(function() {
          $("img").fadeIn(1000);
          $("#picframe").slideToggle("slow");
        });
      });
    </script>
  </body>
</html>

```

In jQuery, it's important to sequence our effects in such a way that they don't run over one another. We'll deal with this issue throughout the book.

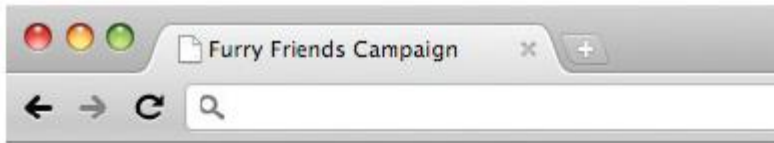
\$("img").fadeIn(1000);  
 \$("#picframe").slideToggle("slow");

We run the fade effect on our image first.

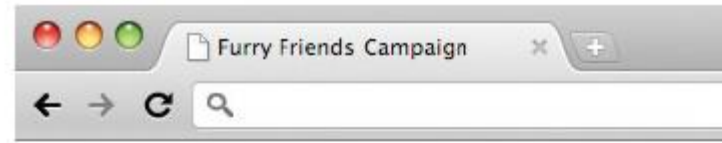
We added some extra stuff in the parentheses to juice up the effects. We'll look at these in more depth in Chapter 5.



# That's it?



<http://ksamkeun.dothome.co.kr/wp/jquery/ch01/ch01-jpg-css.zip>



Your image should  
fade in and slide  
down.

<http://ksamkeun.dothome.co.kr/wp/jquery/ch01/end/>

## 실습과제 18-2

First jQuery gig 예제를 구현하시오.

# Selectors and methods: Grab and go



# Jump for Joy needs your help

You receive an email from your friend, who is a professional portrait photographer.

She wants to roll out a "Jump for Joy" promotion that allows users to win deals on a package of prints. She needs your help making the promotion work.

From: **Emily**  
Subject: **Jump for Joy Promotion!**

Hey,

I saw your tweet that you're doing more interactive web work these days, so I was hoping you could help me with making some interactive stuff for the "Jump for Joy" promotion on my website. I'd like to give my visitors a chance to receive a discount off their purchase before they check out, to encourage them to click around the site some more (and hopefully buy more as a result!).

The page should have four sections with one of four images per section. I need a message that says "Your Discount is" that displays a random discount amount (between 5 and 10 percent). When a user clicks on one of the sections, the message should appear below the image in that section. If a user clicks again, I'd like to get rid of the last message and display a new one.

I've attached a mockup of how I want it to look.

Think you can help??

--

Emily



# What are the project requirements?

To-Do List:

1. The page should have four sections with one of four “jump for joy” images per section.
2. The sections should be clickable.
3. We need a message that says “Your Discount is” along with a random discount amount (between 5 and 10 percent).
4. When a user clicks on one of the sections, the message should appear below the image in that section.
5. If a user clicks again, get rid of the last message and make a new one.

각 섹션 클릭 => 할인을 표시

다시 클릭 => 새 할인을 표시

Add the following items to the page and check them off as you're done:

- A tag to include the jQuery library.
- A <div> tag with the ID of header.
- A <div> tag with the ID of main.
- Inside each of the four div elements inside of the main div, put a different
- image (get the images here: <http://ksamkeun.dothome.co.kr/wp/jquery/ch02/ch02-images.zip>)

```
<html>
  <head>
    <title>Jump for Joy</title>
    <link href="styles/my_style.css" rel="stylesheet">
  </head>
  <body>
    .....
    <h2>Jump for Joy Sale</h2>
  </div>
  .....
  <div></div>
  <div> ..... </div>
  <div> ..... </div>
  <div> ..... </div>
  </div>
  .....
  <script > </script>  </body>
</html>
```



index.html

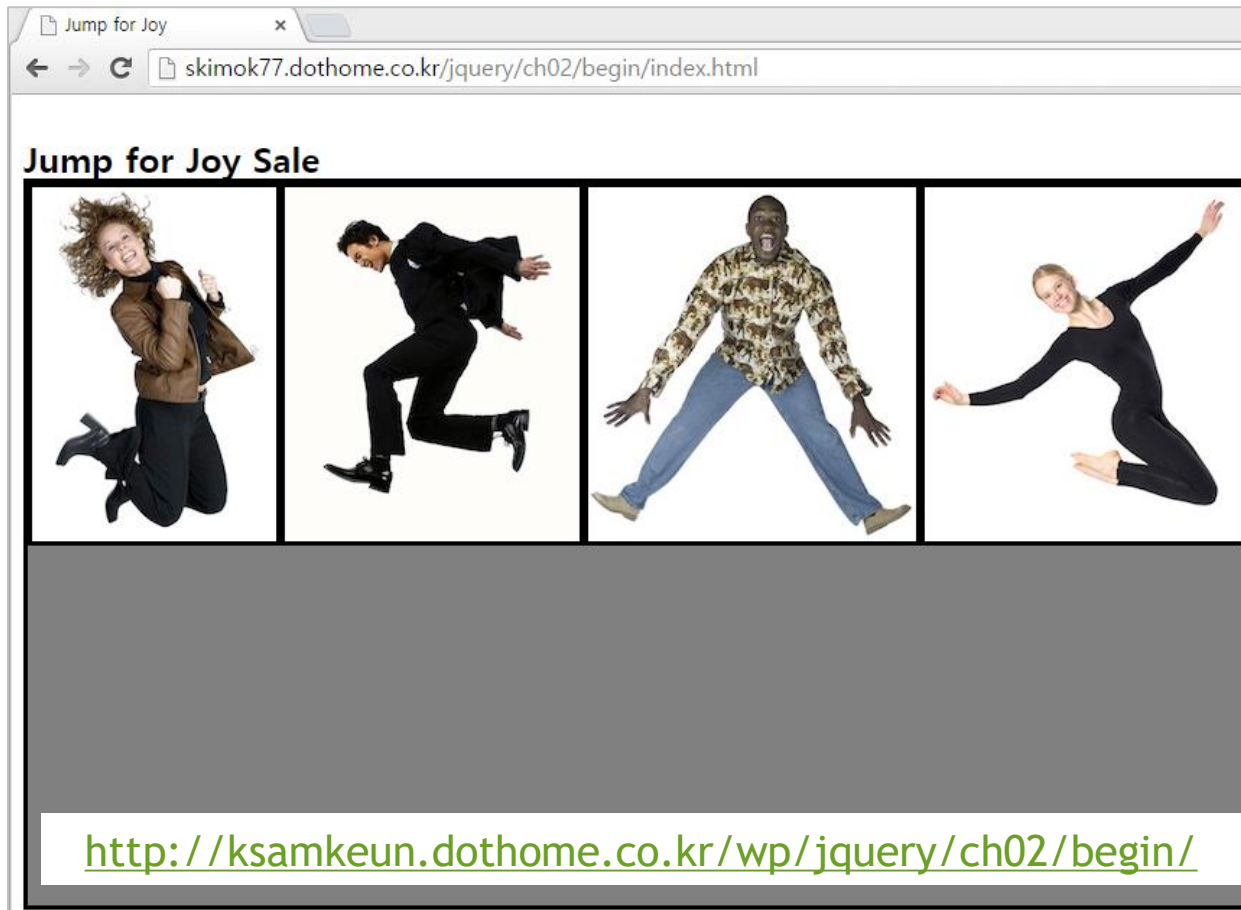
```
div{
  float:left;
  height:245px;
  text-align:left;
  border: solid #000 3px;
}
#header{
  width:100%;
  border: 0px;
  height:50px;
}
#main{
  background-color: grey;
  height: 500px;
}
```



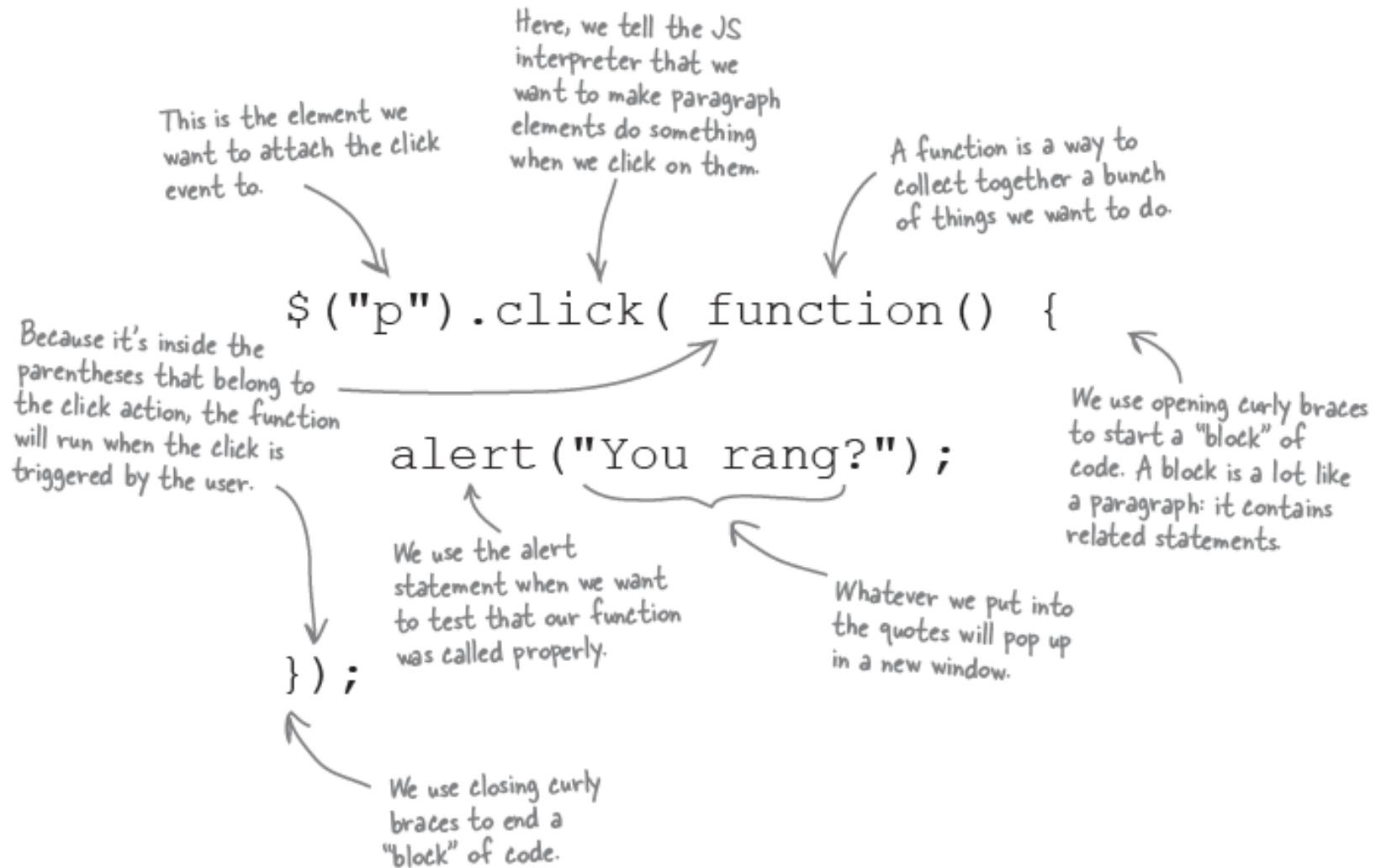
my\_style.css

# 실습과제 18-3 Test Drive

Open the page up in your favorite browser to make sure everything's working. This will give us an opportunity to note how we want the page to function.



# A click event up close



# Add the click method to your page

Update your HTML file to include this script. Don't forget to put it **inside** a `<script>` tag!

The alert function calls up a window in your browser with a message inside it. We'll use it whenever we want to see the results of things we've added to code like variables and functions.

```
<script >
  $(document).ready(function() {
    $("div").click(function() {
      alert("You clicked me.");
    }); //end click function
  }); //end doc ready
</script>
</body>
</html>
```

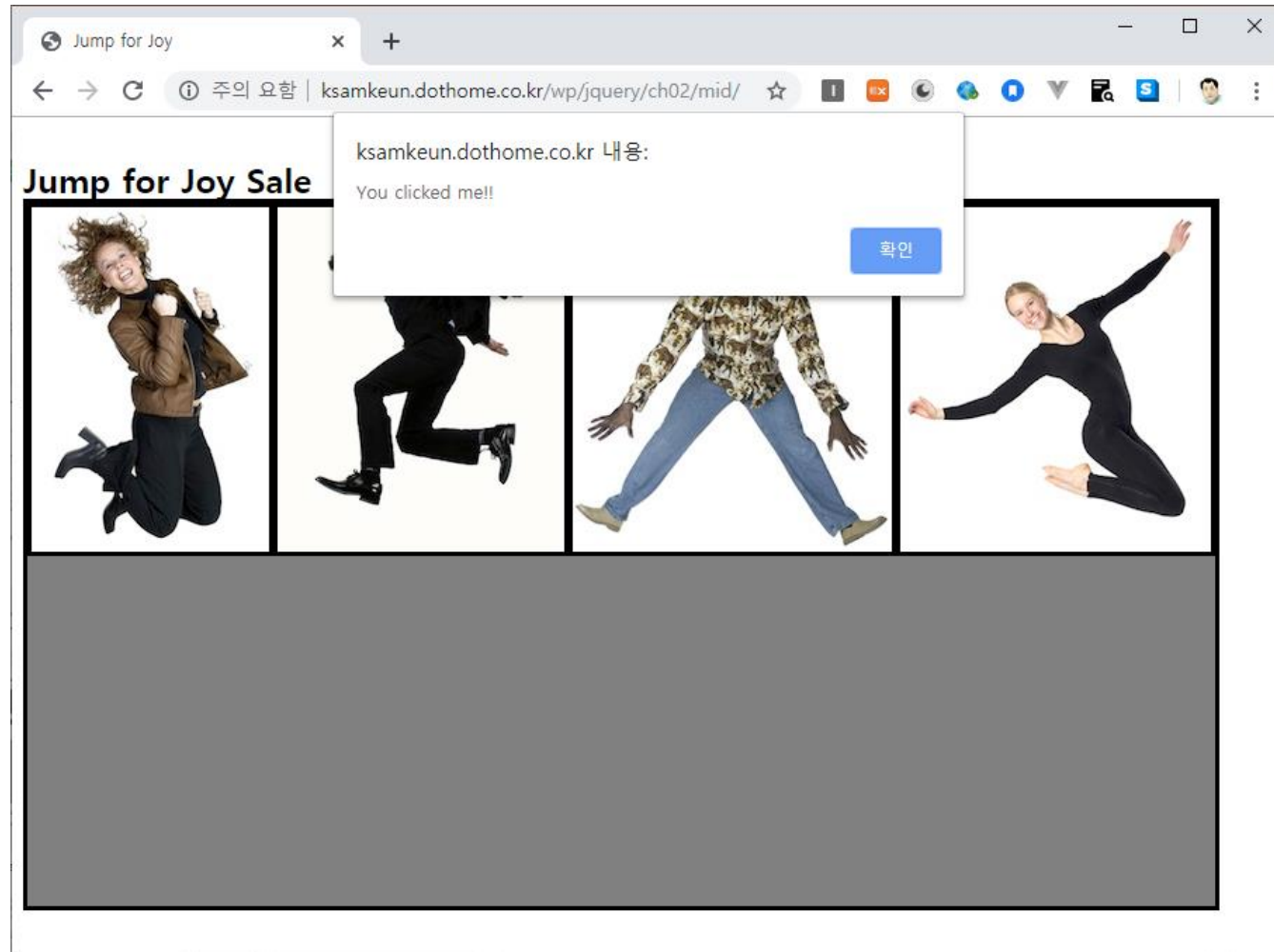
Add these lines between your `<script>` tags to make the divs clickable.

Some programmers add comments to help identify parentheses and curly braces. It's a matter of coding style that's entirely up to you



index.html

# 실습과제 18-4 Test Drive



<http://ksamkeun.dothome.co.kr/wp/jquery/ch02/mid/>





Yes, but no matter *where* I click, I get the alert message. Why is that?

### Hmmm, that is a problem.

It looks like we've gotten a bit click-happy.  
Let's take a look at that click event again.

The JS interpreter did exactly what we asked it to do. It selected all the divs...

...and added a click method to each of them.

`$("div").click( );`

In fact, you don't even have to click on the images to get that message. Our page structure has div elements nested in another div, so when you click on those, the browser thinks you've clicked on both, and you **might get two alerts in that case**.

Clearly, we **need to narrow down** what we're asking jQuery to do here...

# Get more specific

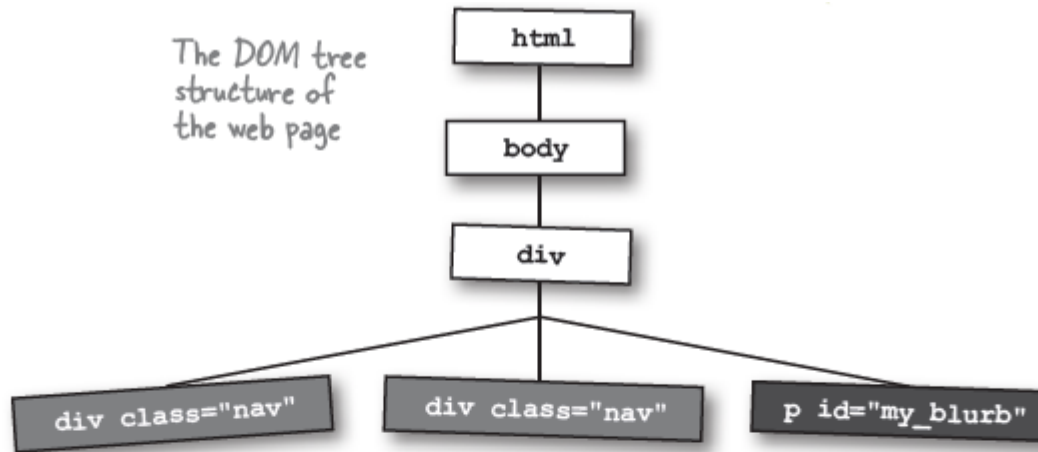
We don't want this div to be clickable. It will be used for display purposes only.

We want each of these divs to be clickable. They'll be used for the interactive functionality.



# Classing up your elements

The DOM tree structure of the web page



엘리먼트에 클래스를 적용해 보자!

Class selectors match any elements that are members of the class.

```
.nav {  
  display: block;  
  border: solid #00f 1px;  
  width: 100%;  
}
```



CSS code

```
$(".nav").click( function() {  
    alert("You clicked  
me!");  
});
```



jQuery code

# ID-entifying elements

An **ID selector** is used to identify a **single, unique element on a page**. In jQuery, as in CSS, the **#symbol** is used to identify an **ID selector**.

IDs are great when you want to get specific with an element, or when there is only going to be one of that kind of element on the page, like a page header or footer.

ID 선택자는 페이지에 단 하나만 있는, 유일한 엘리먼트를 식별할 때 사용!!

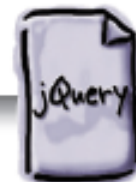
ID selectors match one unique element.

```
#my_blurb {  
  display: block;  
  border: 0px;  
  height: 50%;  
}
```



CSS code

```
$("#my_blurb").slideToggle("slow");
```



jQuery code

# Wire up your web page

Classes and IDs are common ground for the three layers of a web page that we looked at in Chapter 1: **structure**, **style**, and **script**.

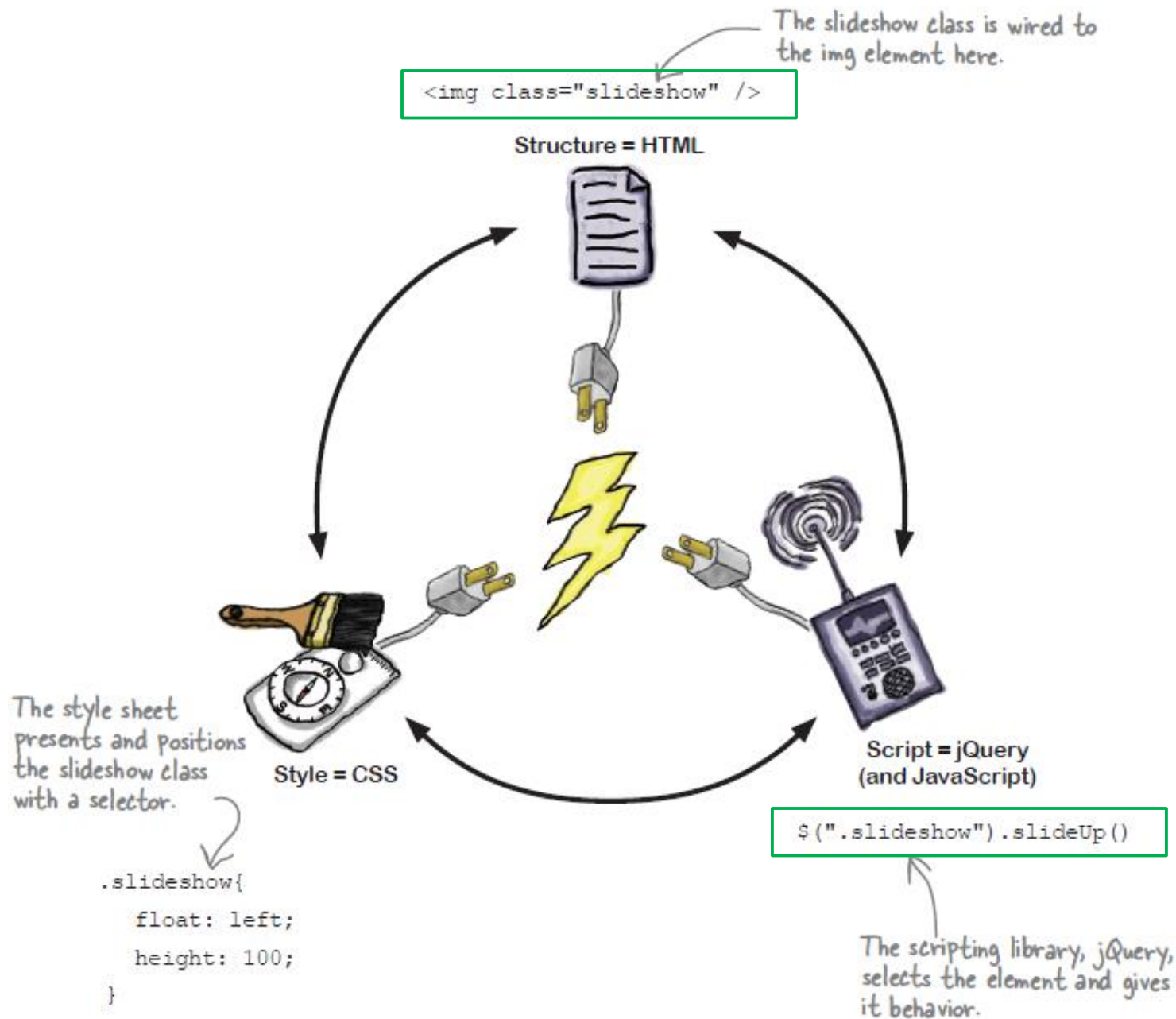
**HTML** provides the building blocks (i.e., elements and their attributes), or **structure** of the web page.

**CSS** provides the **style**, or the presentation and position of those elements.

**JavaScript** and **jQuery** provide the **script** that controls the behavior or function of those elements.

빌딩 블록들을 연결해서 웹 페이지를 만든다!

**.slideUp()** 메소드를 적용할 이미지에 **slideshow**라는 클래스를 지정했다고 하자:



## 실습과제 18-5

Add the **guess\_box** class to all the div elements that will be used to hide the discount code. Also, update our **selector** to use this class, and add it into our CSS file. And it was the main div element that needed to get its ID attribute back.

```
<html>
  <head>
    <title>Jump for Joy</title>
    <link href="styles/my_style.css" rel="stylesheet">
  </head>
  <body>
    <div id="header">
      <h2>Jump for Joy Sale</h2>
    </div>
    <div .....>
      <div .....></div>
      <div .....></div>
      <div .....></div>
      <div .....></div>
    </div>
    <script src="scripts/jquery-1.6.2.min.js"></script>
    <script>
      $(document).ready(function() {
        $(".....").click(function() {
          alert("You clicked me.");
        });
      });
    </script>
  </body>
</html>
```



index.html

```
div{
  float:left;
  height:245px;
  text-align:left;
  border: solid #000 3px;
}
#header{
  width:100%;
  border: 0px;
  height:50px;
}
#main{
  background-color: grey;
  height: 500px;
}
```

height 245px;



my\_style.css

이미지 div 섹션만 클릭될 수 있도록 페이지의 구조와 스타일, 스크립트를 변경하시오.

# Creating some storage space

The var keyword lets you declare a variable.

After the var keyword, you name your variable.

This is how we set the value of the variable in code.

```
var pts = 250;
```

When we declare a variable, the JavaScript interpreter gives us some browser memory in which we can store your data.



We name a variable so that we can reference it later in our script.



We place a value into our variable using an equals sign.





# Mix things up with concatenation

We need to push together (or concatenate) three pieces of information.

"High score: <strong>"

pts

"</strong>"

Which give us:

```
var msg = "High score: <strong>" + pts + "</strong>"
```

When setting a text or HTML value, we use quotes.

When referencing a variable, we use its name without quotes.

We can put HTML tags into variables, too!

The "+" character lets you concatenate (or put together) text, numbers, variables, and much more.

# Meanwhile, back in the code...

Now that you've got a variable set up to store your concatenated **discount** message, you just need to update what's in between your `<script>` tags, so let's focus there.

```
<div class="guess_box">
```

```
<script>
  $(document).ready(function() {

    $(".guess_box").click( function() {

      var discount = Math.floor( (Math.random()*5) + 5 );
      var discount_msg = "<p>Your Discount is "+ discount +"%</p>";
      alert(discount_msg);

    });
  });
</script>
```

Create new JavaScript variables.

We put the discount variable in our alert to make sure it's doing what we want it to.



index.html

# Insert your message with append

```
<p>jQuery lets me add stuff onto my web page  
without having to reload it.</p>
```

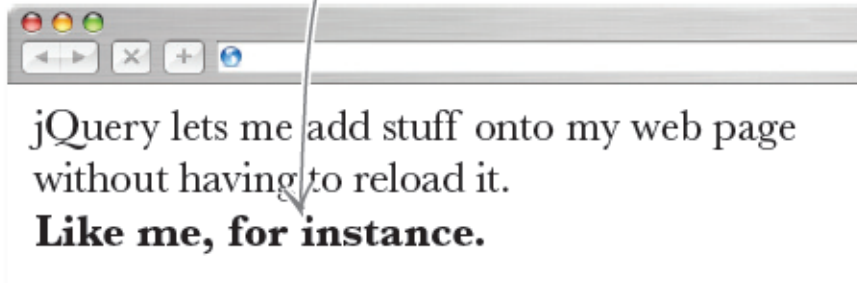


This jQuery statement is telling the JS interpreter to append the content in quotes to all paragraph elements.

```
$("p").append(" <strong>Like me, for instance.</strong>");
```



If you run this in your script, the text in bold appears on your page.



The resulting HTML as seen in the DOM

```
<p>jQuery lets me add stuff onto my web page  
without having to reload it.</p> <strong>Like me,  
for instance.</strong>
```



클릭한 이미지 아래에 메  
시지를 표현하려면?

.append() 메소드 사용

# 실습과제 18-6 Test Drive

Jump for Joy

ksamkeun.dothome.co.kr/wp/jquery/ch02/mid2/

Jump for Joy Sale

ksamkeun.dothome.co.kr 내용:  
<p>Your Discount is 8%</p>

확인



Your Discount is 8%	Your Discount is 8%	Your Discount is 8%	Your Discount is 8%
Your Discount is 7%	Your Discount is 7%	Your Discount is 7%	Your Discount is 7%
Your Discount is 8%	Your Discount is 8%	Your Discount is 8%	Your Discount is 8%

<http://ksamkeun.dothome.co.kr/wp/jquery/ch02/mid2/>

# Everything works great, but...

```
<script>
$(document).ready(function() {
  $(".guess_box").click(function() {
    var discount = Math.floor((Math.random()*5) + 5);
    var discount_msg = "<p>Your Discount is "+discount+"%</p>";
    alert(discount_msg);
    $(".guess_box").append(discount_msg);
  });
});
</script>
```

This applied the click method so that each member of the guess\_box class is clickable.

This is just to test the variable.

Our selector is specific enough to grab a class, but we end up affecting all the divs in the class

클릭한 바로 그 div만 할인 메시지를 추가해야 한다.

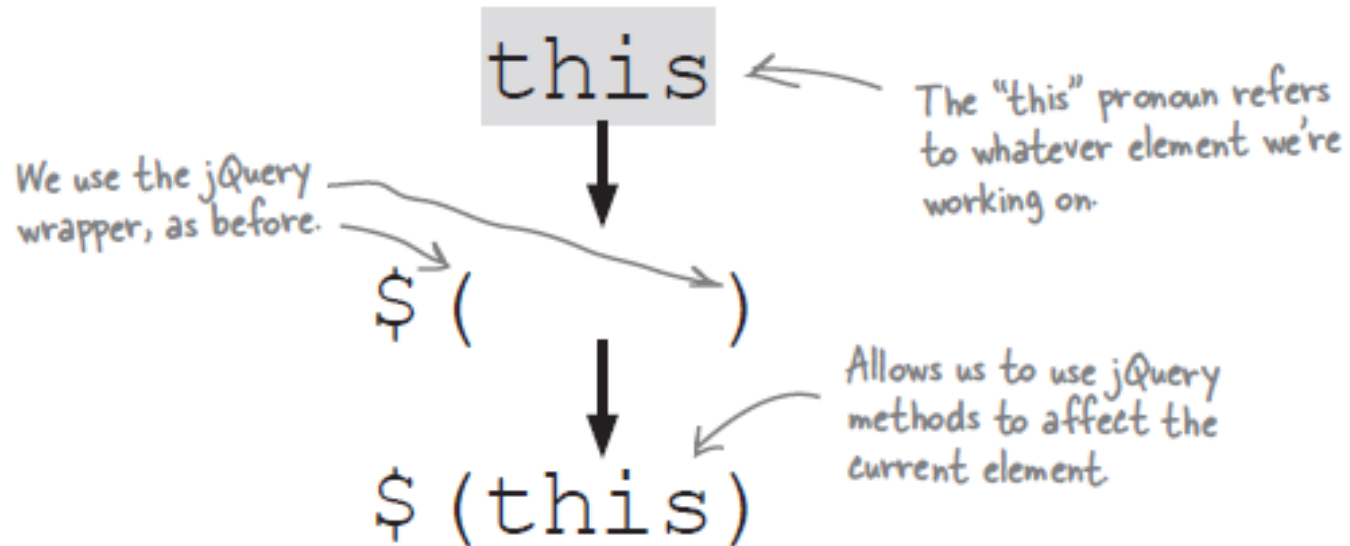
⇒ 클릭한 div만 선택하고, 할인 메시지를 그 div에만 추가하려면?

Wouldn't it be dreamy if there were a  
simple way to select the div we clicked?  
But I know it's just a fantasy...



# Give me `$(this)` one

The `$(this)` selector gives us an easy way to point to the **current** element.



It's important to think about `$(this)` as **context-dependent**.

In other words, `$(this)` means different things **depending on where or when you use it**.

One of the best places to use it is within a function that runs **when a jQuery method is called**:

The diagram illustrates the context of `$(this)` within a jQuery `click` event handler. It shows the following code structure with handwritten annotations:

```
$( "#myImg" ).click( function() {  
    $(this).slideUp();  
} );
```

Annotations and their targets:

- "Here's the selector to access our element" points to `"#myImg"`.
- "Call a jQuery method." points to `.click`.
- "Run this function when the method is called." points to the opening curly brace of the function `{`.
- "Both click and slideUp are jQuery methods. You know you're dealing with a method or a function when you see the parentheses." points to the closing curly brace of the function `}`.
- "Access the current element (#myImg, in this case) inside our function." points to `$(this)`.



# Put \$(this) to work

Let's see if **\$(this)** can help us solve our problem.


Update your code to use **\$(this)**, as shown in bold below.

```
<script type="text/javascript">
  $(document).ready(function() {

    $(".guess_box").click( function() {

      var discount = Math.floor((Math.random()*5) + 5);
      var discount_msg = "<p>Your Discount is "+ discount +"%</p>";
      alert(discount_msg);
      $(this).append(discount_msg);

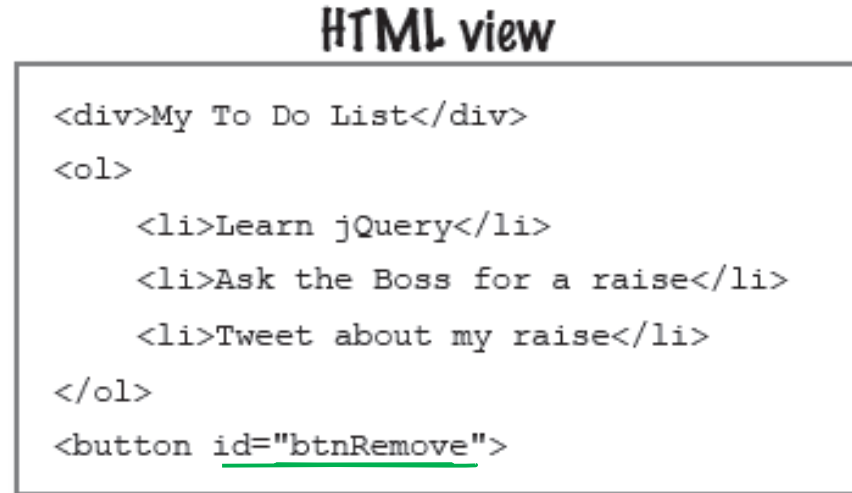
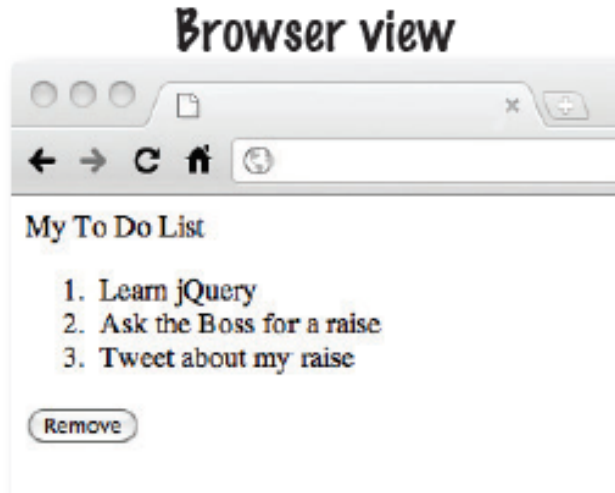
    });
  }); //end doc ready
</script>
```



Now we're telling our guess boxes to append the discount code only to the one clicked.

# Good riddance with **remove**

1. Here's what it looks like in the browser, and the HTML that creates it.



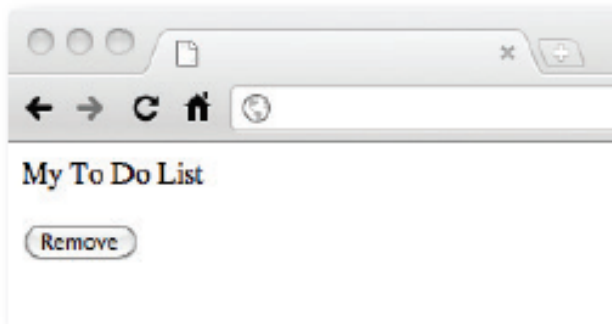
2. And here's the code for the button, which will remove all the list items from your list:

```
$("#btnRemove").click(function() {
  $("li").remove();
});
```

remove is another  
jQuery method. Think  
of a jQuery method as  
a verb—it's all about  
web page action.

- Looking again at the page in the browser and the HTML—after jQuery is finished—we can see that all our list items are gone, even in the HTML!

### Browser view



### HTML view

```
<div>My To Do List</div>
<ol>
</ol>
<button id="btnRemove">
```

# Dig down with descendant selectors

왼쪽 선택자가 부모

`$ ("div p")`

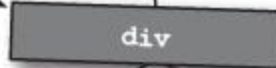
오른쪽 선택자가 자식

Leave a space between the parent element name and child element name.



This returns all div elements that are children of a div element.

`$ ("div div")`



Grabs all img elements that are children of the children of a div element

`$ ("div div img")`



(Would these be grand-divs then?)

자식의 자식을 선택하는 방법

`$ ("div div p")`



When you combine class and ID selectors with descendant selectors, you can really get specific, which works great on a complex HTML page.

`$ ("div p#my_blurb")`



`$ ("div p")`


Grabs all p elements that are children of a div element

# 실습과제 18-7 Test Drive

Jump for Joy

ksamkeun.dothome.co.kr/wp/jquery/ch02/end/

## Jump for Joy Sale



Your Discount is 6%

<http://ksamkeun.dothome.co.kr/wp/jquery/ch02/end/>

# Q & A

