CORSO DI LAUREA IN INFORMATICA

PROGRAMMAZIONE WEB

JQUERY

a.a 2017-2018

Cos'è

- jQuery (http://jquery.com/) è una libreria di classi e funzioni Javascript che permette al programmatore di compiere in modo facile quanto segue:

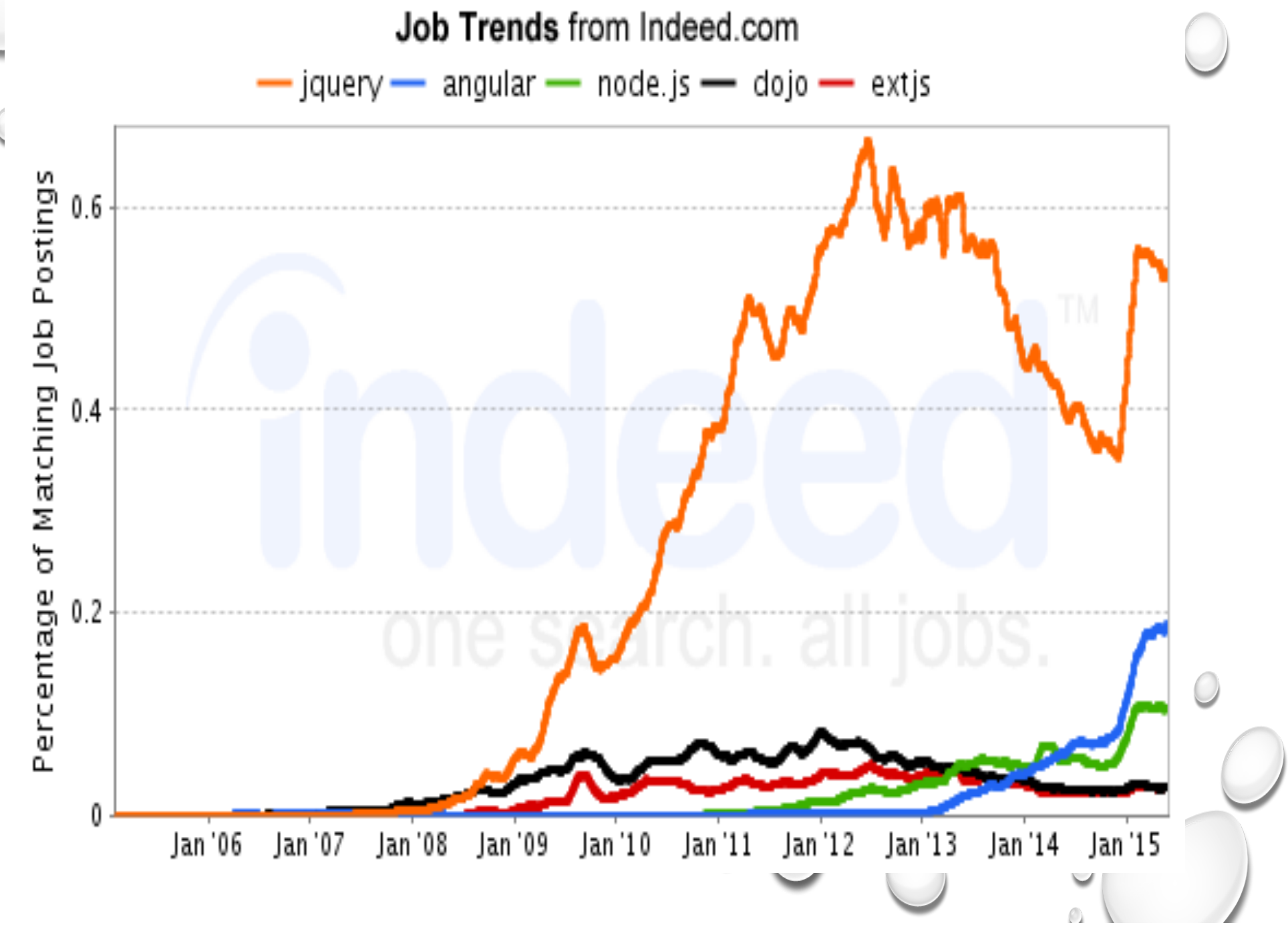
– Navigazione nel documento HTML,

– gestione degli eventi,
– animazioni,
– funzionalità AJAX.

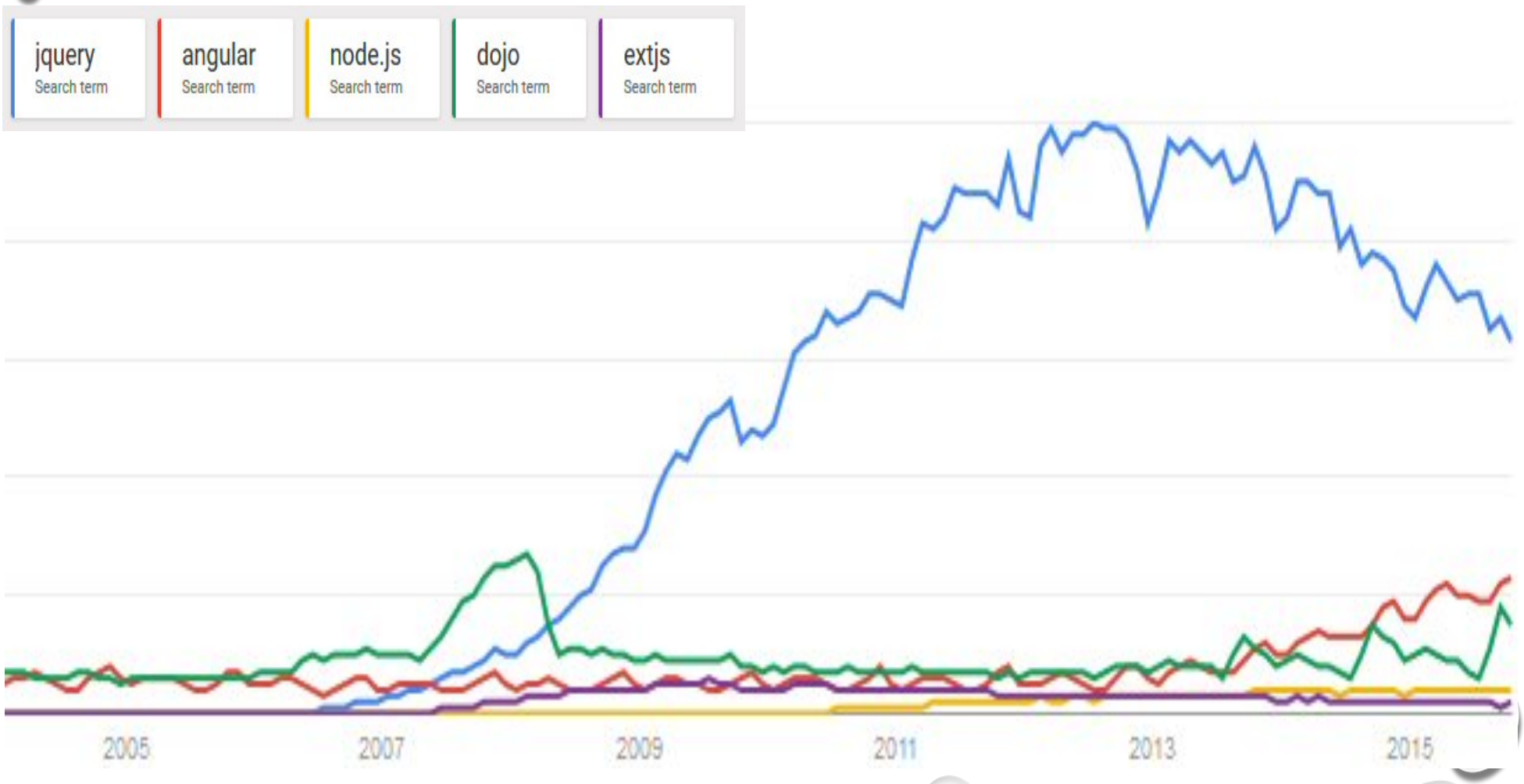
Perché è utile?

- Consente di scrivere codice in modo più compatto e ad alto livello.
- Funge da “normalizzatore”, fornendo dei costrutti conformi agli standard del W3C, indipendentemente dal browser su cui gira.

Industry Usage (Job Postings)



INDUSTRY USAGE (GOOGLE SEARCHES)



Browser Compatibility

- Chrome, firefox, opera
  - Current and previous major version
    - Older versions tend to work, but are not tested on new jquery code
    - Same strategy as google docs
- Internet explorer
  - Jquery 1.9: IE 6 and later
  - Jquery 2.X: IE 9 and later (same API as jquery 1.9)
    - As of 10/2015, no mention of microsoft edge on official support page, but work is in progress
- Safari
  - 5.1 and later
- Android browser
  - 4.0 and later
- Ios browser
  - 6.1 and later

JavaScript Testing

- Problem (from first section on general javascript)
  - Java: very strict compliance tests to be called "java"
    - You can have very high confidence that code written in java 8 on windows version will run identically (except for some differences in how guis look) on java 8 on macos, linux, solaris, and other windows versions. True for java from oracle, apple, IBM, or open source version from brazil.
  - Javascript: every browser vendor does it themselves, with no outside checks
    - Behavior of same javascript program can vary substantially from browser to browser, and even from one release of the same browser to another
- Consequence
  - Before final deployment, you must test on all browsers you expect to support
  - One of main benefits of jquery is that it tries to hides browser differences, and it mostly succeeds
    - But even so, you must test on full range of browsers before final deployment

Download

- • Scaricare la libreria da http://docs.jquery.com/Downloading_jQuery
- • Due release:
  – compressa (per siti in “produzione”)

    (http://code.jquery.com/jquery-1.12.3.min.js),
- – non compressa (per sviluppo)

    (http://code.jquery.com/jquery-1.12.3.js).

- Rinomina il file con un nome generico
  - Rename file to jquery.js (or possibly jquery-2.js)
    - Lets you switch from jquery-2.1.4.js to jquery-2.1.4.min.js without editing many HTML files
    - Similarly, lets you later upgrade to 2.1.5 without editing many HTML files

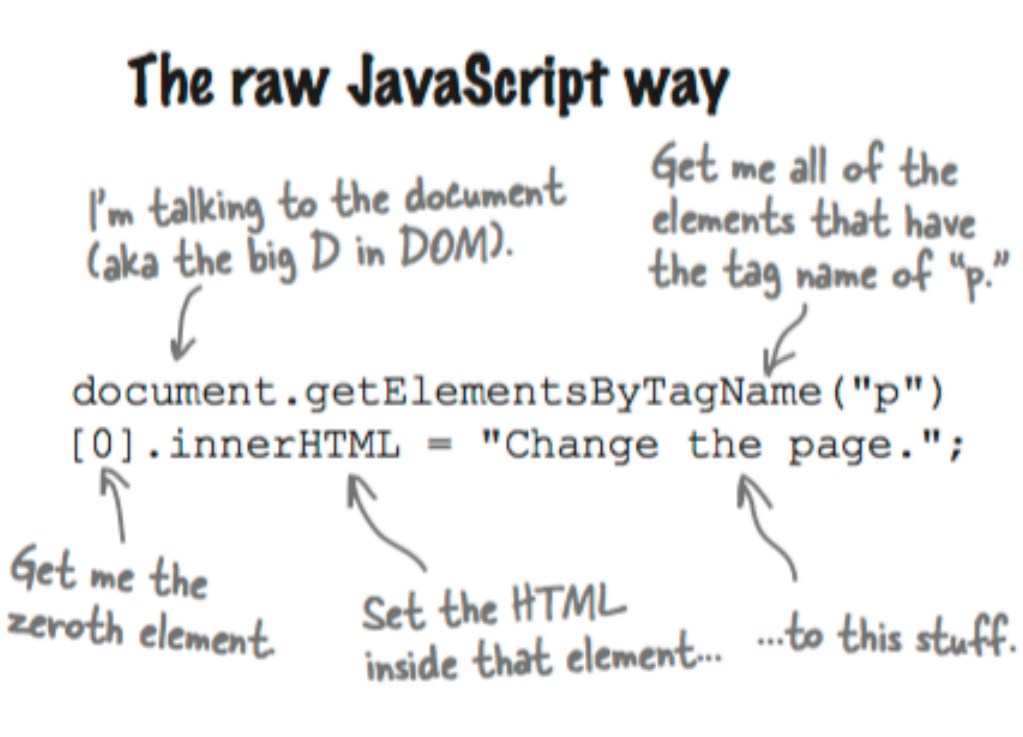
Typical Approach for Loading jQuery

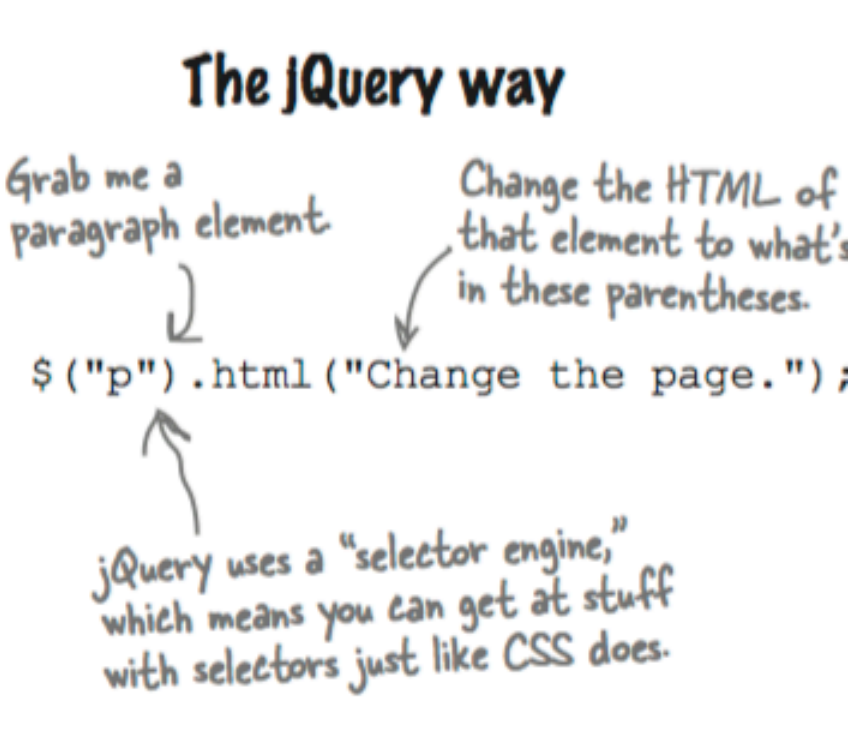
```
...
<Head>
<Title>your title</title>
<Link rel="stylesheet"
      Href="css/your-styles.Css"/>
<script src="scripts/jquery.Js"></script>
<Script src="scripts/your-
 script.Js"></script>
</Head>
...
```

- You should load jQuery before loading your own scripts that make use of jQuery.
- You should rename jquery-x.y.z.js to jquery.js.

Jquery intro

- jQuery *is* JavaScript, but a much more approachable version. When you want to control the DOM, jQuery makes it much easier.





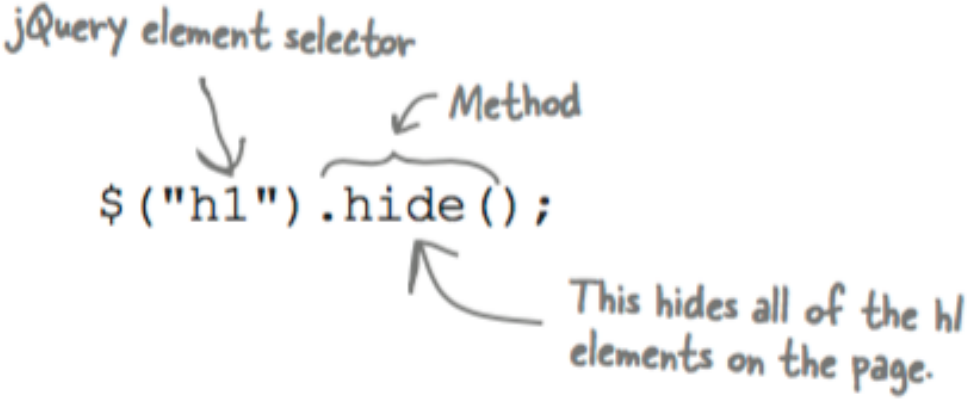
jQuery selects elements the same way CSS does

CSS selector

Element selector

```
h1 {
      text-align: left;
}
```

jQuery selector



CSS selector

Class selector

```
.my_class{
      position: absolute;
}
```

ID selector

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

jQuery selector

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

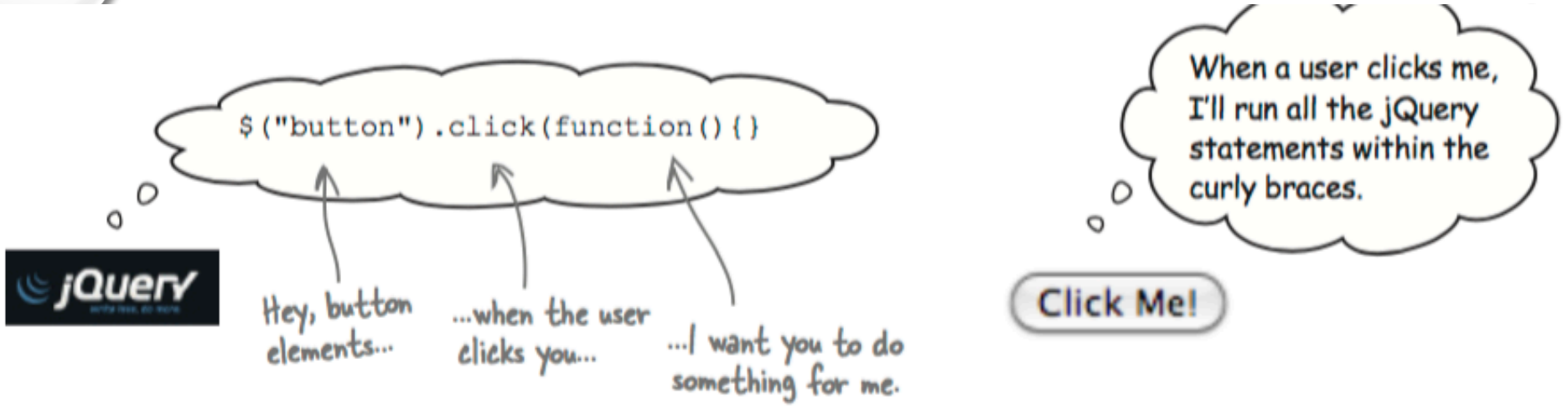
```
$("#my_id").fadeOut();
```

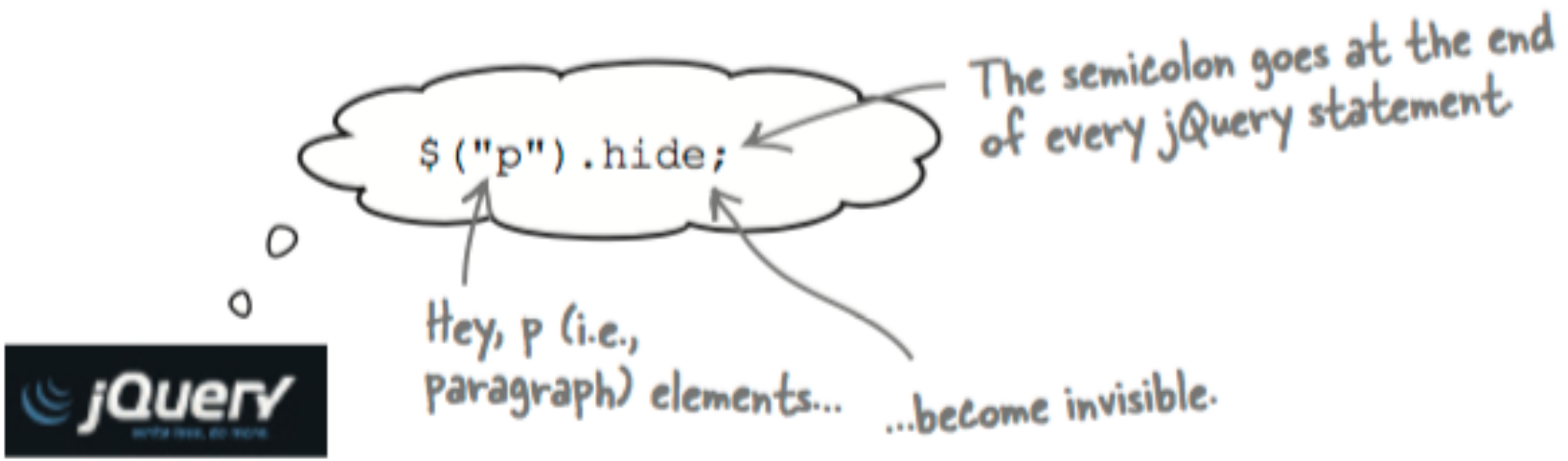
And this jQuery statement fades out an element that has a CSS ID of my_id until it's invisible.

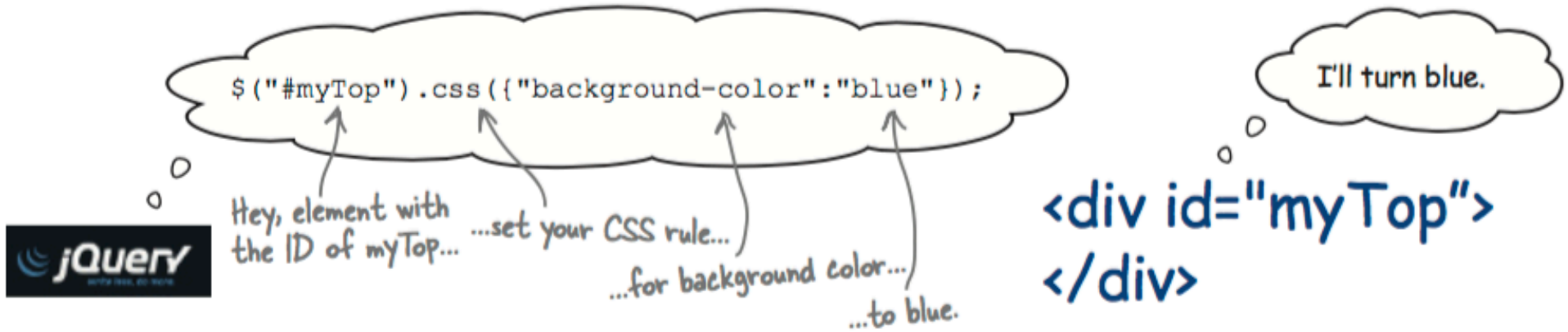
CSS selectors select elements to add style to those elements; jQuery selectors select

elements to add behavior to those elements.

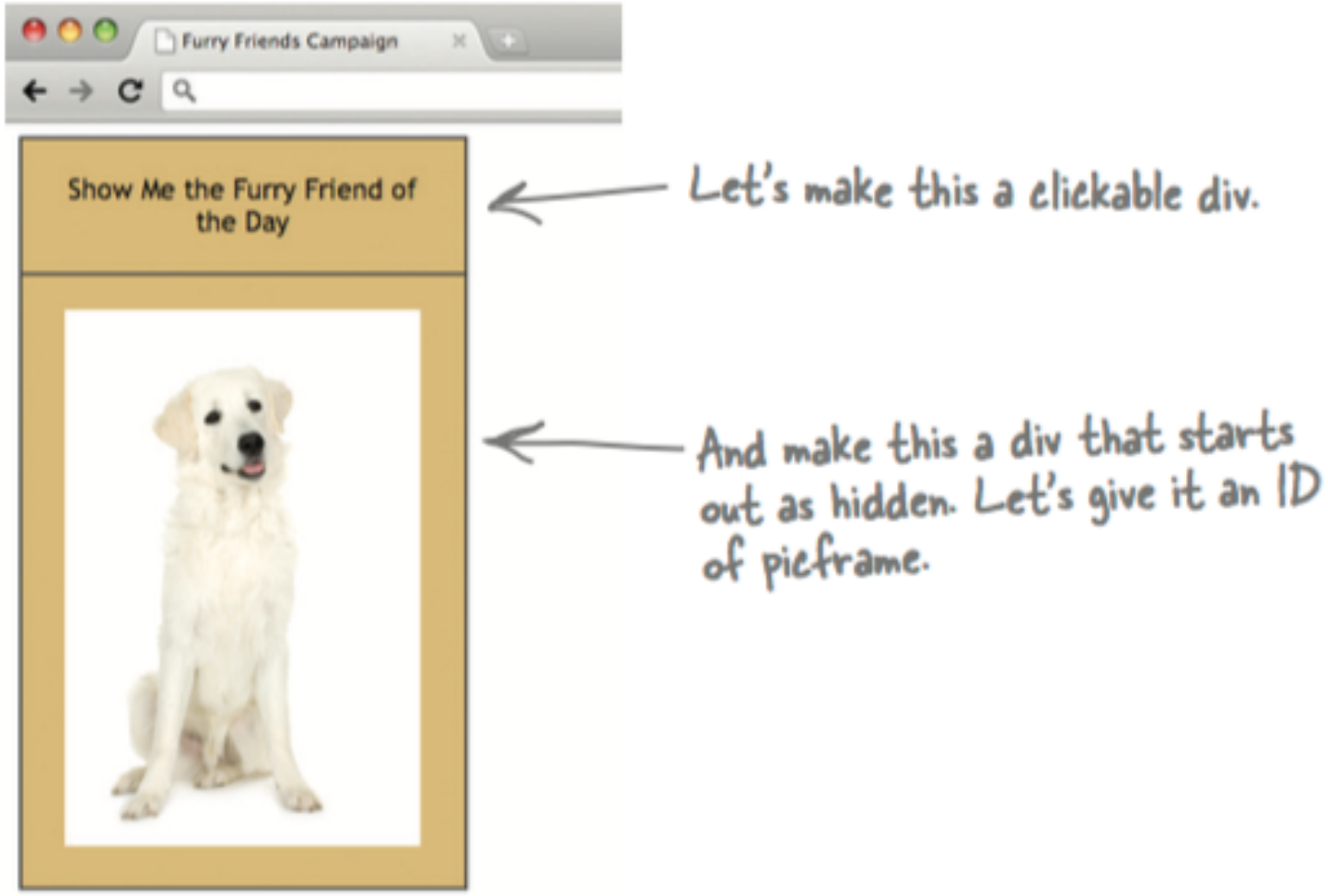
JQuery in translaction






$("#myTop").css({"background-color":"blue"});
Hey, element with the ID of myTop...
...set your CSS rule...
...for background color...
...to blue.
jQuery
I'll turn blue.

Our Furry Friends Need Your Help



Things to Try First

- "$" Is the name of the main jquery function (with alias "jquery")

- $("H1")
  - If you have at least 1 h1 in the page, this should return an array of the matches. If you have no h1's, this should return an empty array. In general, you can supply a CSS selector pattern as the argument to $.
- $("some-css-pattern").Hide("slow")
  - First, try

    `$("some-css-pattern");`

    if it returns a non-empty array, try

    `$("some-css-pattern").Hide("slow");`

    and watch the matching elements disappear. Use

    `$("some-css-pattern").Show("slow");`

    to make them come back.

Example (code 01-begin)

```
<!DOCTYPE html>
<html>
  <head>
    <title>Furry Friends Campaign</title>
    <link rel="stylesheet" type="text/css" href="styles/my_style.css">
  </head>
  <body>
    <div id="showfriend">
      <a href="#">
        <h2>Our Furry Friends Need Your Help</h2>
        <img src="images/furry_friend.jpg" alt="Our Furry Friend">
      </a>
    </div>
  </body>
</html>
```

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">
     <img src="images/furry_friend.jpg">
    </div>
    <script src="scripts/jquery-1.6.2.min.js"></script>
    <script>
      $(document).ready(function(){
         $("#clickMe").click(function() {

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

Nest the furry_friend.jpg image inside the picframe.

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

index.html

my_style.css

```
#clickMe {
    background: #D8B36E;
    padding: 20px;
    text-align: center;
    width: 205px;
    display: block;
    border: 2px solid #000;
}
#picframe {
    background: #D8B36E;
    padding: 20px;
    width: 205px;
    display: none;
    border: 2px solid #000;
}
```

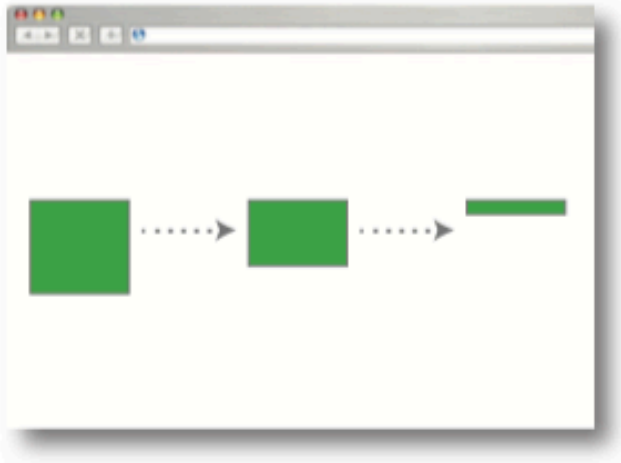
This styles the clickMe div so that so it has the same look and feel as the picframe div.

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

my_style.css

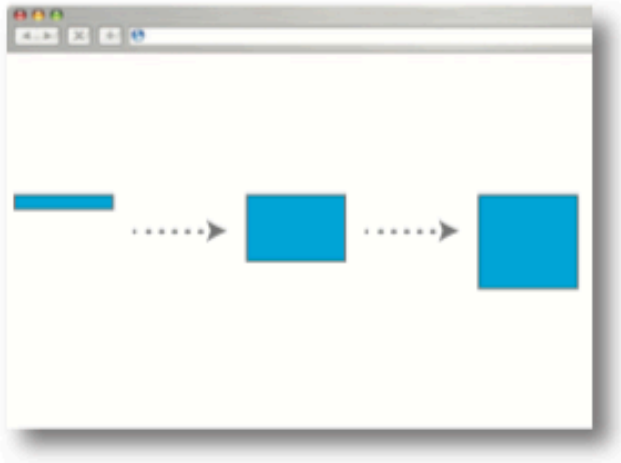
Slide on in...

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



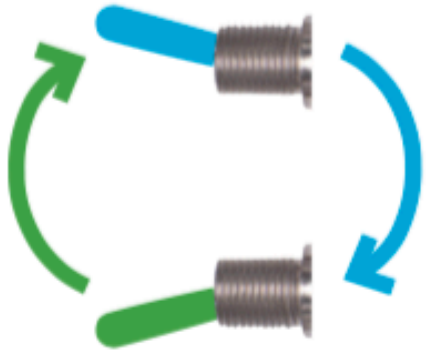
The slideUp method changes the height property of the element until it's 0, and then hides the element.

```
$("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."

Slide on in (2)

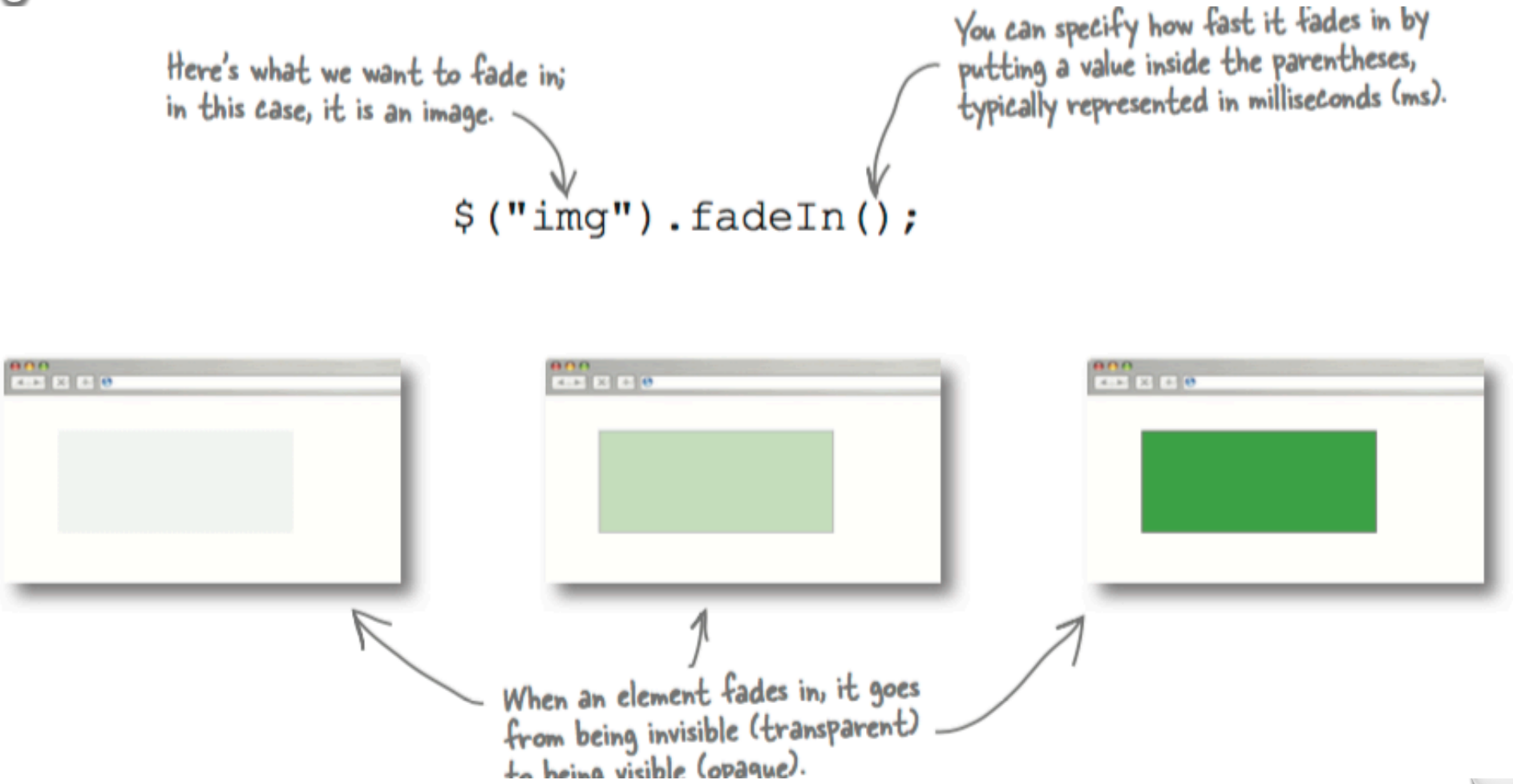
- To hide/show an element:

```
$("p.foo") .slideUp(1000, function(){ console.log("Nascosto!"); })
.slideDown(1000, function(){ console.log("Visibile!"); });
```

- **$("p.foo") .slideToggle(1000, function(){ console.log("Invertito!"); });**

Fade

- the image gradually appears, going from invisible to fully visible.



Index.html

```
<body>
  <div id="clickMe">Show me the Furry Friend of the Day</div>
  <div id="picframe">
  <img src="images/furry_friend.jpg">
  </div>
  <script src="scripts/jquery-1.6.2.min.js"></script>
  <script>
    $(document).ready(function(){
      $("#clickMe").click(function() {
        $("img").fadeIn(1000);
        $("#picframe").slideToggle("slow");
      });
    });
  </script>
</body>
```

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.

We run the fade effect on our image first.

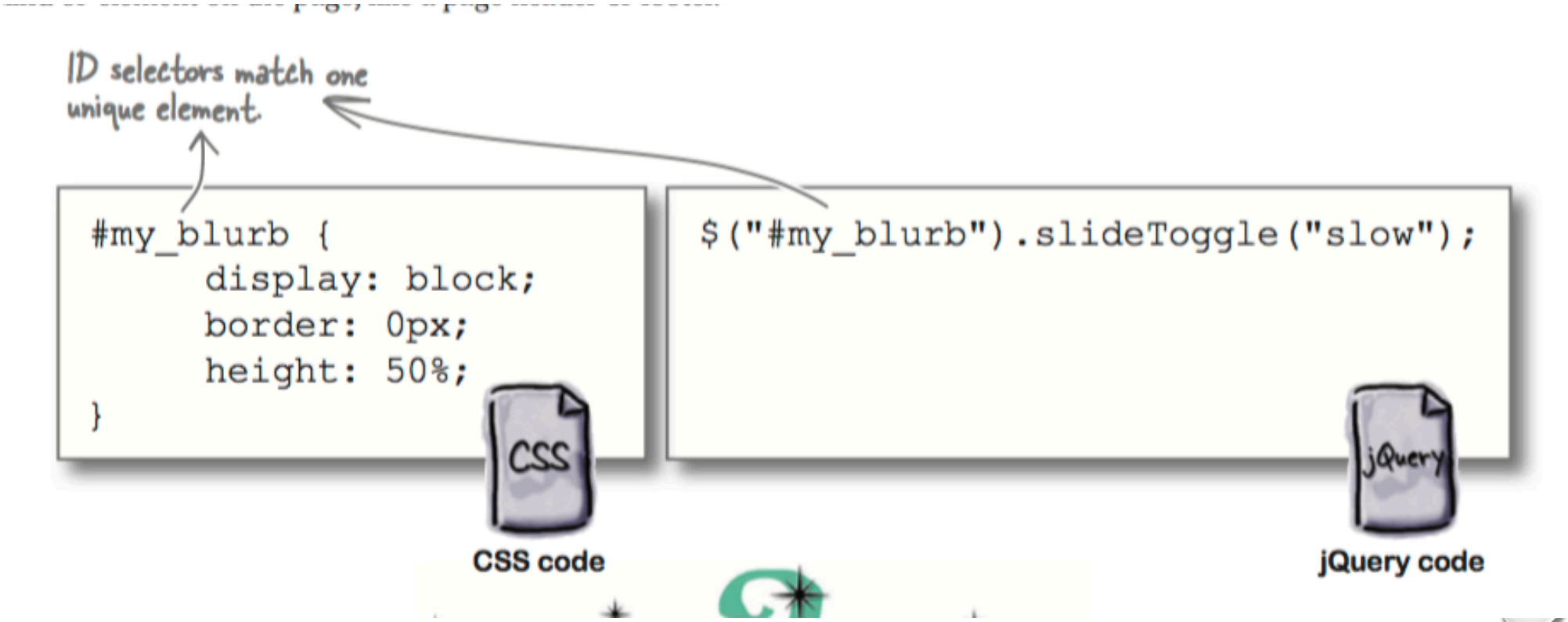
We added some extra stuff in the parentheses to juice up the effects. We'll look at these in more depth in Chapter 5.

Idea

- Manipulating the DOM
  - One of the main uses of jquery is to find elements in the DOM (document object model – the tree structure that represents the HTML page) and modify them in various ways

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.



SHOW/HIDE

- Per nascondere il paragrafo con id bar:
  - **$("#bar").hide();**
- 'elemento rimane tuttavia presente nel DOM e per tornare a visualizzarlo è sufficiente eseguire:
  - **$("#bar").show();**
- Esempio più complesso:
  - **$("#bar")**

.css({ "background":"yellow",

"border":"1px solid black" }) .hide(2000, function() { console.log("Animazione completata!"); });

Manipulating DOM: Overview

- Find HTML elements that match a pattern
  - $(“css selector pattern")
    - Returns array of HTML elements that match
- Perform operations on the elements
  - $(“css selector pattern").Op1(…);
    - Single operation
  - $("Css selector pattern").Op1(…).Op2(…).Op3(…);
    - Consecutive operations via “chaining”
- Example
  - $("Div h3").Addclass("yellow").Hide("slow");
    - Finds all h3’s that are inside a div, adds the CSS class named “yellow”, then slowly makes them disappear

Selecting DOM Elements: Simple Examples

- $("#Some-id")
  - Return 1-element set (or empty set) of element that has that id
  - Simplest use, and very common for ajax (note the "#")
- $("p")
  - Return all p elements
- $(".Blah")
  - Return all elements that have class="blah"
- $("li p span.Blah")
  - Return all elements that are inside p elements, that in turn are inside li elements

Manipulating DOM Elements: Commonly Used Functions

- $("#Some-id").Val()
  - Returns value of input element. Used on 1-element sets.
- $("Selector").Each(function)
  - Calls function on each element. “This” set to element.
- $("Selector").Addclass("name")
  - Adds CSS class name to each. Also removeclass, toggleclass
- $("Selector").Hide()
  - Makes invisible (display: none). Also show, fadeout, fadein, etc.
- $("Selector").Click(function)
  - Adds onclick handler. Also change, focus, mouseover, etc.
- $("Selector").Html("<tag>some html</tag>")
  - Sets the innerhtml of each element. Also append, prepend

Events

- An event represents the precise moment when something happens.
- Examples:
- moving a mouse over an element
- selecting a radio button
- clicking on an element
- Example:
- **$(document).ready()**
- The $(document).ready() method allows us to execute a function when the document is fully loaded.

Events (2)

- **click()**
- The click() method attaches an event handler function to an HTML element.
- The function is executed when the user clicks on the HTML element.
- The following example says: When a click event fires on a <p> element; hide the current <p> element:

```
$("p").click(function(){
    $(this).hide();
});
```

Events(3)

- **dblclick()**
- The dblclick() method attaches an event handler function to an HTML element.
- The function is executed when the user double-clicks on the HTML element:

```
$("p").dblclick(function(){
    $(this).hide();
});
```

Event (4)

- **mouseenter()**
- The mouseenter() method attaches an event handler function to an HTML element.
- The function is executed when the mouse pointer enters the HTML element:

```
$("#p1").mouseenter(function(){
    alert("You entered p1!");
});
```

- **mouseleave()**
- The mouseleave() method attaches an event handler function to an HTML element.
- The function is executed when the mouse pointer leaves the HTML element:

Event (5)

- **mousedown()**
- The mousedown() method attaches an event handler function to an HTML element.
- The function is executed, when the left, middle or right mouse button is pressed down, while the mouse is over the HTML element
- **mouseup()**
- The mouseup() method attaches an event handler function to an HTML element.
- The function is executed, when the left, middle or right mouse button is released, while the mouse is over the HTML element:

Event (6)

- h**over()**
- The hover() method takes two functions and is a combination of the mouseenter() and mouseleave() methods.
- The first function is executed when the mouse enters the HTML element, and the second function is executed when the mouse leaves the HTML element:

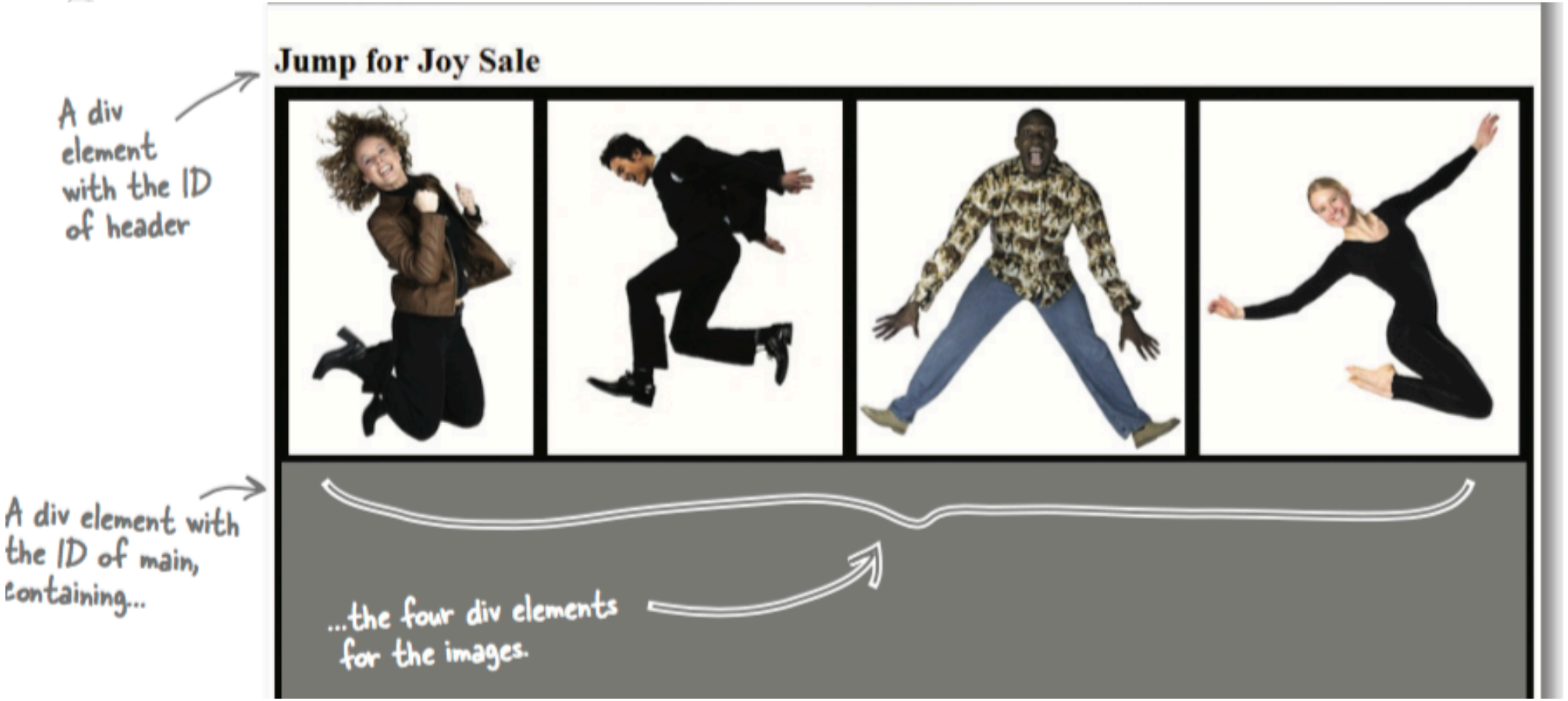
```
$("#p1").hover(function(){
    alert("You entered p1!");
},
function(){
    alert("Bye! You now leave p1!");
});
```

Code 01 Jquery_DOM_Fun

```
<!DOCTYPE html>
<html><head> <title>jQuery Meets the DOM</title>
<style>
#change_me {position: absolute; top: 100px; left: 400px; font: 24px arial;}
#move_up #move_down #color #disappear {padding: 5px;}
</style>
 <script src=scripts/jquery-1.5.2.min.js></script>
</head>
<body>
  <button id="move_up">Move Up</button>
  <button id="move_down">Move Down</button>
  <button id="color">Add Color</button>
  <button id="disappear">Disappear/Re-appear</button>

  <div id="change_me">Make Me Do Stuff!</div>
<script>
  $(document).ready(function() {
    $("#move_up").click( function() {
        $("#change_me").animate({top:30},200);
    });//end move_up
    $("#move_down").click( function() {
        $("#change_me").animate({top:500},2000);
    });//end move_down
    $("#color").click( function() {
        $("#change_me").css("color", "purple");
    });//end color
    $("#disappear").click( function() {
        $("#change_me").toggle('slow');
    });//end disappear
```

Example code 02 begin



Index.html

```
<!DOCTYPE html>
<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 id="main">
      <div><img src="images/jump1.jpg"/></div>
      <div><img src="images/jump2.jpg"/></div>
      <div><img src="images/jump3.jpg"/></div>
      <div><img src="images/jump4.jpg"/></div>
    </div>
    <script src="scripts/jquery-1.6.2.min.js"></script>
    <script>
    </script>
  </body>
</html>
```

Index.html

- Update the structure, style, and script in your page to make only the four image div sections clickable. In the CSS file, create a CSS class (called guess_box) and apply it to the html and the script.

```
<div id="main">
    <div class="guess_box"><img src="images/jump1.jpg"/></div>
    <div class="guess_box"><img src="images/jump2.jpg"/></div>
    <div class="guess_box"><img src="images/jump3.jpg"/></div>
    <div class="guess_box"><img src="images/jump4.jpg"/></div>
</div>
```

ll css

```css
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;
}
```

```css
div{
  float:left;
  border: solid #000 3px;
  text-align:left;
}

.guess_box{
  height:245px;
}

#header{
  width:100%;
  border: 0px;
  height:50px;
}

#main{
  background-color: gray;
  height: 500px;
}
```

This (01 /this.html)

```
<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>
```

Jquery CSS Classes

- JQuery has several methods for CSS manipulation. We will look at the following methods:

addClass() - Adds one or more classes to the selected elements

removeClass() - Removes one or more classes from the selected elements

toggleClass() - Toggles between adding/removing classes from the selected elements

css() - Sets or returns the style attribute

Example JQuery classes (2)

- You can also specify multiple classes within the addClass() method:

```
$("button").click(function(){
    $("#div1").addClass("important blue");
});
```

- The following example shows how to remove a specific class attribute from different elements:

```
$("button").click(function(){
    $("h1, h2, p").removeClass("blue");
});
```

```
<!DOCTYPE html>
<html>
<head>
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.3.1/jquery.min.js">
</script>
<script>
$(document).ready(function(){
    $("button").click(function(){
        $("p:first").addClass("intro note");
    });
});
</script>
<style>
.intro {
    font-size: 30px;
    color: blue;
}
.note {
    background-color: yellow;
}
</style>
</head>
<body>

<h1>This is a heading</h1>

<p>This is a paragraph.</p>
<p>This is another paragraph.</p>

<button>Add two class names to the first p element</button>

</body>
</html>
```

01/addclass.html

Filters

- The three most basic filtering methods are first(), last() and eq(), which allow you to select a specific element based on its position in a group of elements.
- Other filtering methods, like filter() and not() allow you to select elements that match, or do not match, a certain criteria.

Filters: first()

- The **first()** method returns the first element of the selected elements.
- The following example selects the first <p> element inside the first element:

```
$(document).ready(function(){
    $("div p").first();
});
```

http://www.w3schools.com/jquery/tryit.asp?filename=tryjquery_first

Filters: last()

- The **last()** method returns the last element of the selected elements.
- The following example selects the last <p> element inside the last element:

```
$(document).ready(function(){
    $("div p").last();
});
```

- http://www.w3schools.com/jquery/tryit.asp?filename=tryjquery_last

Filters: eq()

- The **eq()** method returns an element with a specific index number of the selected elements.
- The index numbers start at 0, so the first element will have the index number 0 and not 1.
- The following example selects the second <p> element (index number 1):

```
$(document).ready(function(){
  $("p").eq(1);
});
```

Filters: filter() and not()

- The **filter()** method lets you specify a criteria. Elements that do not match the criteria are removed from the selection, and those that match will be returned.
- The following example returns all <p> elements with class name "intro":

```
$(document).ready(function(){
    $("p").filter(".intro");
});
```

- The **not()** method returns all elements that do not match the criteria.
- The following example returns all <p> elements that do not have class name "intro":

```
$(document).ready(function(){
    $("p").not(".intro");
});
```

Creare elementi nel DOM

- Creare un nuovo paragrafo (notare la presenza delle parentesi angolate):
  - **$(" ");**
- A questo punto possiamo aggiungere attributi e testo al nuovo paragrafo:

$(' Questo è un nuovo paragrafo! ');

- Sintassi alternativa:

$(" ", { "class":"bat", "text":"Questo è un nuovo paragrafo!" });

Si noti che il nuovo elemento non è ancora visibile perché non è stato ancora inserito nel DOM.

Inserire elementi nel DOM (1)

- **append()** e **prepend()** aggiungono, in coda ed in testa rispettivamente, gli elementi passati come argomenti internamente all'elemento a cui vengono concatenati.

```
$("p").append("Some appended text.");
```

http://www.w3schools.com/jquery/tryit.asp?filename=tryjquery_html_append

```
var paragrafo = $("<p>", { "css":{"background":"yellow"}, "text":"Questo è un nuovo paragrafo!" });

$("body").prepend(paragrafo);
```

Esempio Append

```
function appendText() {
    var txt1 = "<p>Text.</p>";              // Create element with HTML
    var txt2 = $("<p></p>").text("Text.");   // Create with jQuery
    var txt3 = document.createElement("p");  // Create with DOM
    txt3.innerHTML = "Text.";
    $(”body").append(txt1, txt2, txt3);        // Append the new elements
}
```

http://www.w3schools.com/jquery/tryit.asp?filename=tryjquery_html_append2

Inserire elementi nel DOM (2)

- **appendTo()** e **prependTo()**:

$("<p>", { "css":{"background":"yellow"}, "text":"Questo è un nuovo paragrafo!" }).prependTo("body");

after() e **before()**: come **append()** e **prepend()**, ma aggiungono il contenuto fuori dall'elemento a cui vengono concatenati.

http://www.w3schools.com/jquery/tryit.asp?filename=tryjquery_html_after2

- **insertAfter()** e **insertBefore()** stanno a **after()** e **before()** come **appendTo()** e **prependTo()** stanno a **append()** e **prepend()**.

Inserire elementi nel DOM (3)

- **wrap()** serve a racchiudere elementi esistenti con nuovi elementi:

– **$("span").wrap("");**

• **unwrap()**: rimuove i tag che racchiudono l'elemento a cui viene concatenato.

- **wrapAll()**: svolge la stessa azione di **wrap()**, ma applicata a più elementi contemporaneamente.
- **wrapInner()**: racchiude il contenuto di un elemento, ma non l suoi tag.

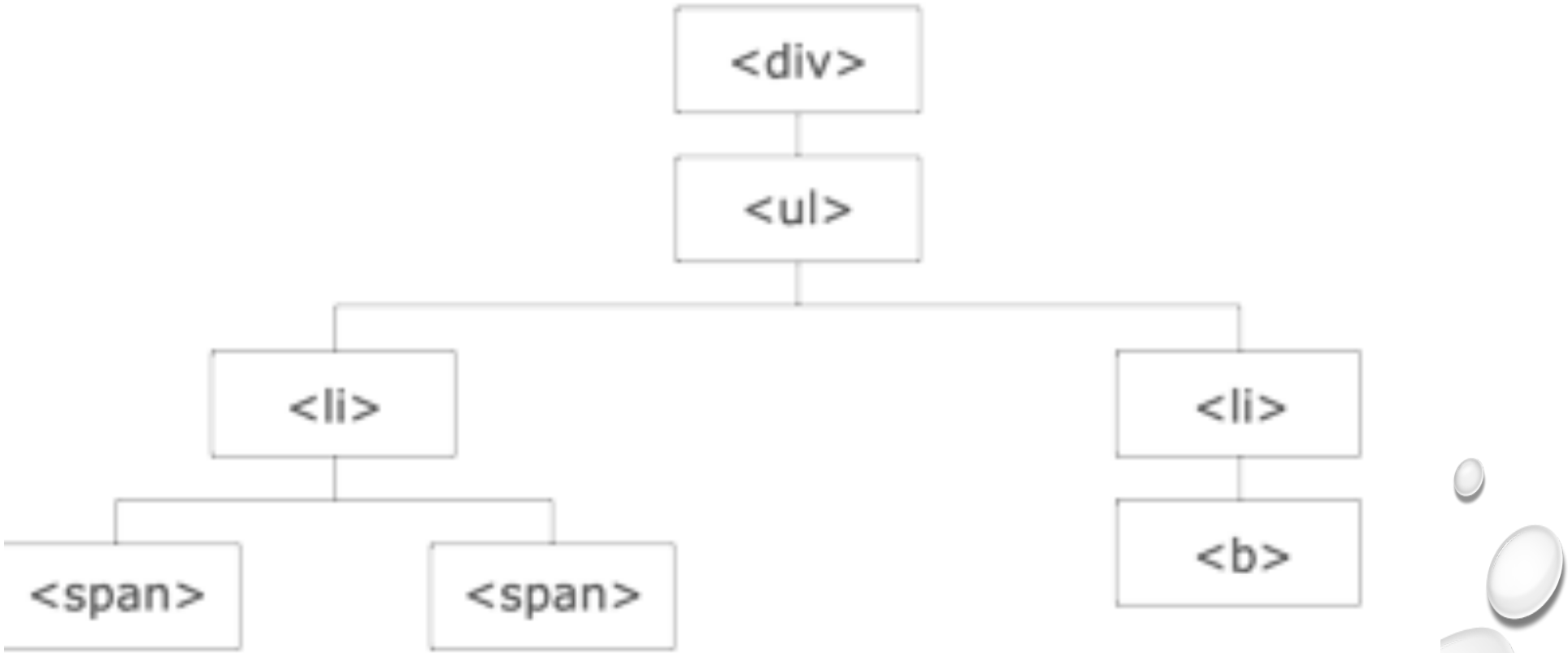
Rimuovere elementi dal DOM

- • Il metodo **remove()** consente di rimuovere gli elementi selezionati (ed i loro figli) dal DOM:
- – **$("p").remove(".foo");**
  Il codice precedente rimuove dal documento tutti i paragrafi con classe **foo.**

- http://www.w3schools.com/jquery/tryit.asp?filename=tryjquery_dom_remove

Jquery traversing

- It is used to "find" (or select) HTML elements based on their relation to other elements. Start with one selection and move through that selection until you reach the elements you desire.



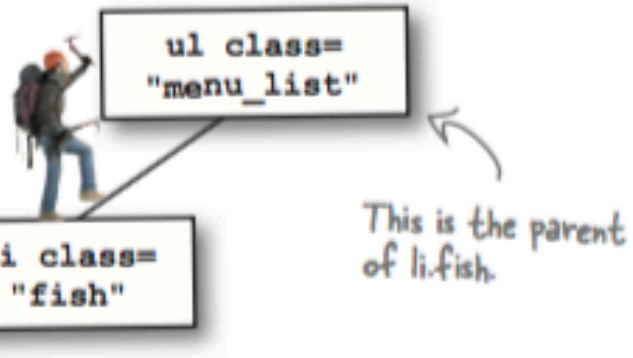
- The element is the **parent** of <ul>, and an **ancestor** of everything inside of it
- The <ul> element is the **parent** of both <li> elements, and a **child** of
- The left <li> element is the **parent** of , **child** of <ul> and a **descendant** of
- The element is a **child** of the left <li> and a **descendant** of <ul> and
- The two <li> elements are **siblings** (they share the same parent)
- The right <li> element is the **parent** of <b>, **child** of <ul> and a **descendant** of
- The <b> element is a **child** of the right <li> and a **descendant** of <ul> and
- An ancestor is a parent, grandparent, great-grandparent, and so on.
- A descendant is a child, grandchild, great-grandchild, and so on.
- Siblings share the same parent.

Climb the DOM

Select all the elements in the fish class.

Then get the element above those elements.

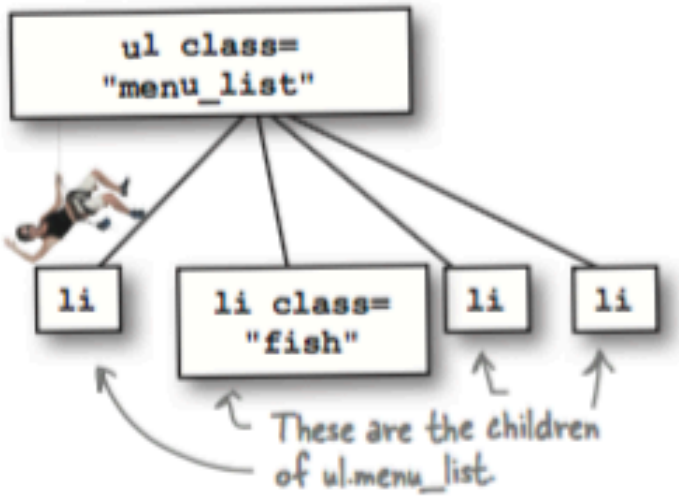
```
$(".fish").parent()
```



Select all the elements in the menu_list class.

Then get the element below those elements.

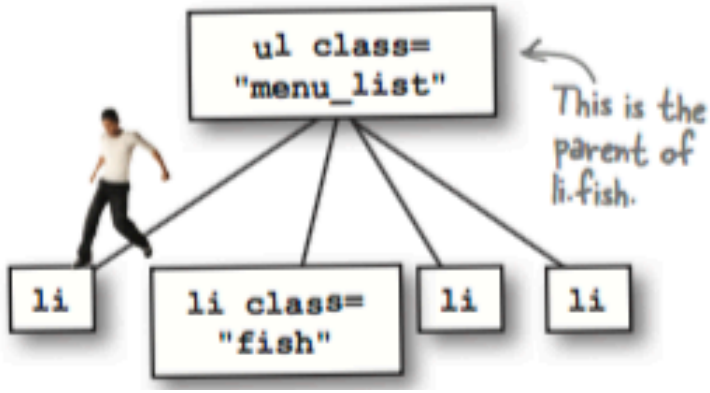
```
$(".menu_list").children()
```



Select all the elements in the fish class.

Then get the sibling element immediately to the left

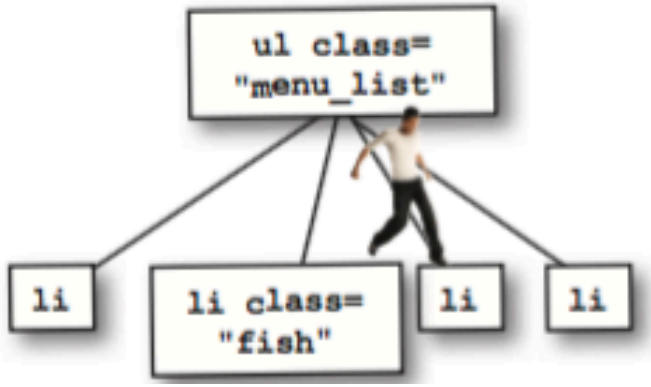
```
$(".fish").prev()
```



Select all the elements in the fish class.

Then get the sibling element immediately to the right

```
$(".fish").next()
```



Esempi

- parent()

http://www.w3schools.com/jquery/tryit.asp?filename=tryjquery_parent

- parents()
- http://www.w3schools.com/jquery/tryit.asp?filename=tryjquery_parents
- You can also use an optional parameter to filter the search for ancestors.

The following example returns all ancestors of all elements that are <ul> elements:

http://www.w3schools.com/jquery/tryit.asp?filename=tryjquery_parents2

jQuery Traversing - Siblings

- Siblings share the same parent.
- JQuery methods for traversing sideways the DOM tree:
  - siblings()
  - next()
  - nextAll()
  - nextUntil()
  - prev()
  - prevAll()
  - prevUntil()

siblings()

- The **siblings()** method returns all sibling elements of the selected element.

```
$(document).ready(function(){
    $("h2").siblings();
});
```

The following example returns all sibling elements of <h2> that are <p> elements:

```
$(document).ready(function(){
    $("h2").siblings("p");
});
```

The **next()** method returns the next sibling element of the selected element.

```
$(document).ready(function(){
    $("h2").next(); //the next sibling of <h2>
});
```

http://www.w3schools.com/jquery/tryit.asp?filename=tryjquery_next

JQuery children() and find() Methods

- The **children()** method returns all direct children of the selected element.

```
$(document).ready(function(){
    $("div").children();
});
```

http://www.w3schools.com/jquery/tryit.asp?filename=tryjquery_children

- The **find()** method returns descendant elements of the selected element, all the way down to the last descendant.
- The following example returns all elements that are descendants of :

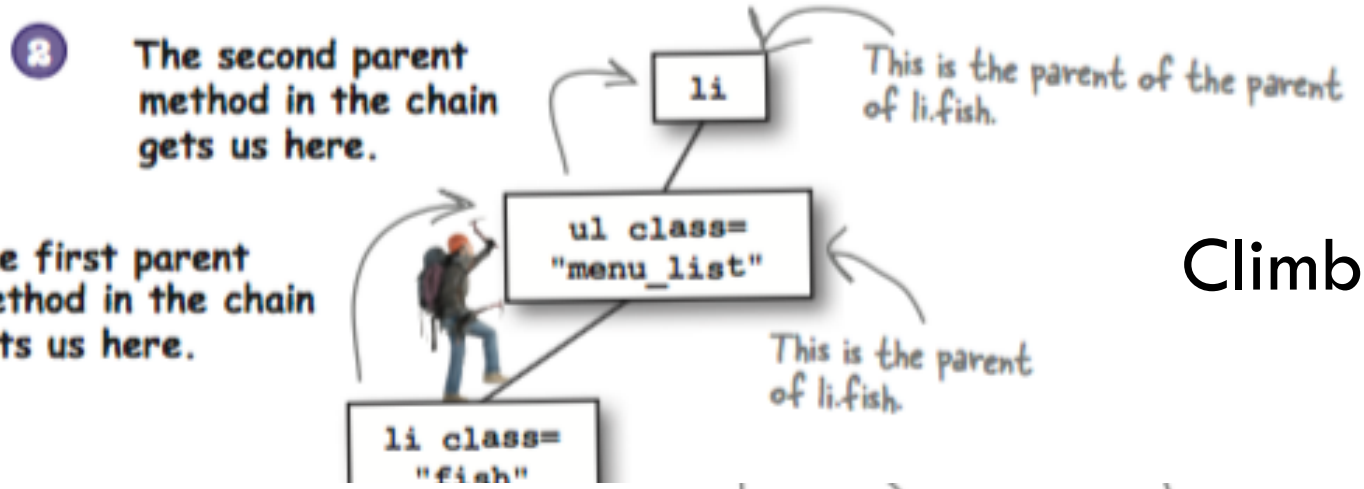
```
$(document).ready(function(){
    $("div").find("span");
});
```

http://www.w3schools.com/jquery/tryit.asp?filename=tryjquery_find

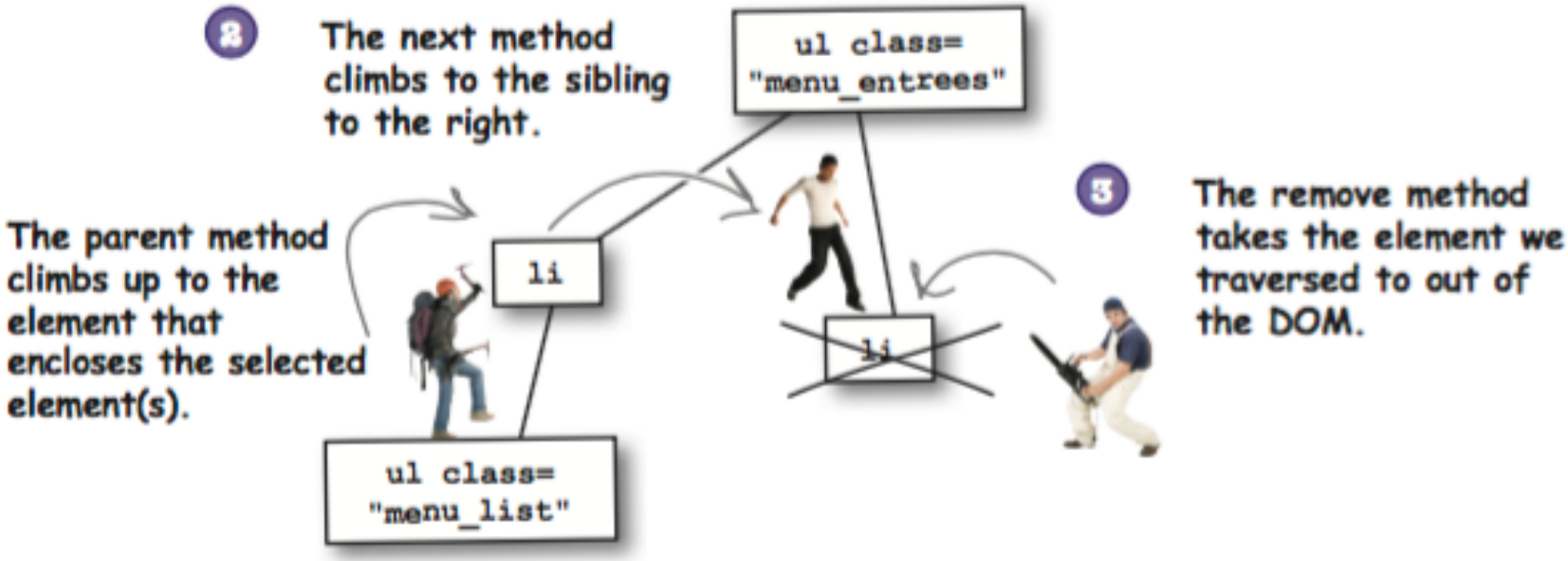
Climb the DOM (2)

```
$(".fish").parent().parent()
```

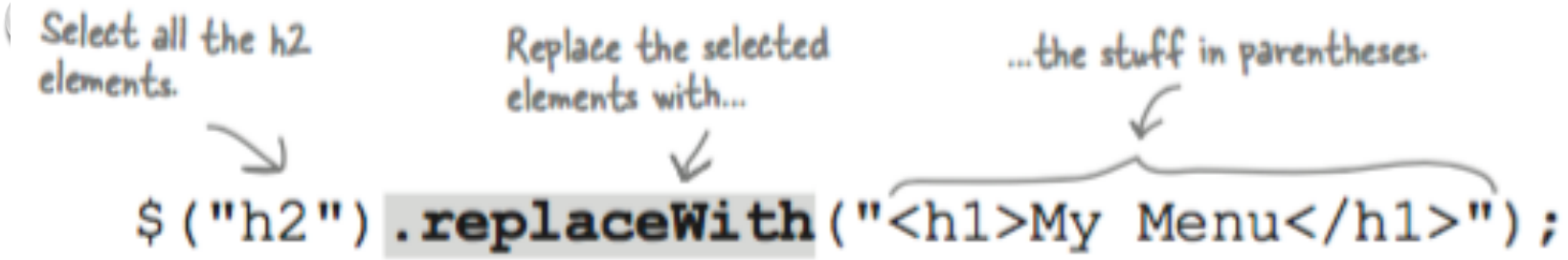
to the method chain.




```
$(".menu_list").parent().next().remove()
```



Change out elements with replaceWith



The replaceWith method works well when you have a one-to-one replacement like exchanging the hamburger class for portbello class. (one-to-one substitution)

Esempio (03): changing menu entry

- We want a “Go Vegetarian" button that automatically substitutes the vegetarian options on our web page menu.
- Here's how our substitutions work:

— We offer no substitutes for our fish entrées, so we need those removed.

— We offer giant portobello mushrooms as a vegetarian substitute for our hamburgers.

— We offer tofu as a vegetarian substitute for all of our meat and egg dishes except hamburgers.

— We'll need a button that restores the menu to its original state.

P.S. If you can pull it off, we’d also like a leaf icon to show up next to the substituted vegetarian entrees.

- One-to-many substitution
- We have to replace many *different* kinds of ingredients (i.e., turkey, eggs, steak, lamb chops) with one ingredient (tofu).
- Many-to-one substitution
- When we want to put the different kinds of meat back in, the DOM has forgotten about them.
- We could replace tofu with just *one* of the types of meat, but that's not what we wanted at all.
- **Insert li elements of the tofu class into the DOM after the meat elements.**
- **Detach the elements of the meat class and hold them in a variable.**

Menu example

We put li.tofu elements into the DOM after the meat elements.

```
$(".meat").after("<li class='tofu'><em>Tofu</em></li>");
```

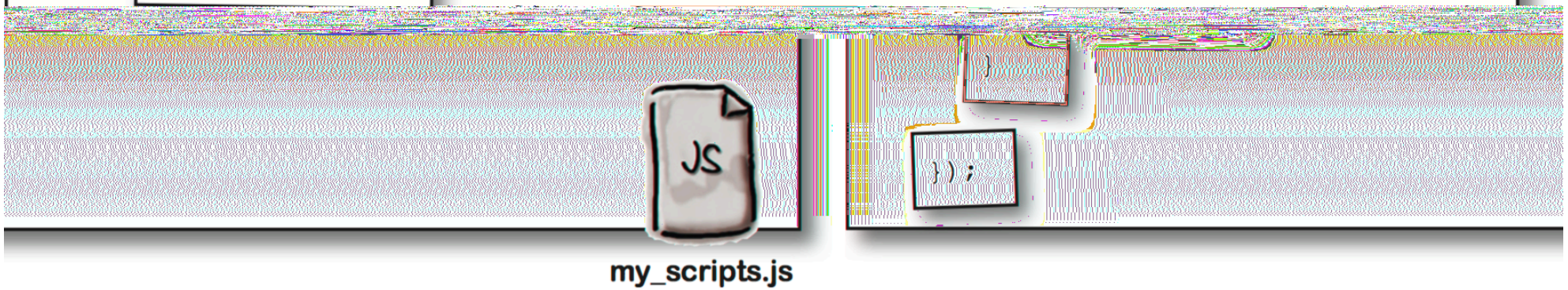
Then, we detached the li.meat elements but held on to them by saving them into $m.

```
$m = $(".meat").detach();
```

```
$("button#restoreMe").click(function(){

    if (v == true){
```

Restore



my_scripts.js

JQuery animation (1)

- The jQuery **animate()** method is used to create custom animations.
- **Syntax:**

  $(*selector*).animate({*params*},*speed*,*callback*);

- The required params parameter defines the CSS properties to be animated.
- The optional speed parameter specifies the duration of the effect. It can take the following values: "slow", "fast", or milliseconds.
- The optional callback parameter is a function to be executed after the animation completes.
- The following example demonstrates a simple use of the animate() method; it moves a element to the right, until it has reached a left property of 250px

```
$("button").click(function(){
    $("div").animate({left: '250px'});
        });
```

- http://www.w3schools.com/jquery/tryit.asp?filename=tryjquery_animation1

Animation: Animate multiple properties

```
$("button").click(function(){
    $("div").animate({
        left: '250px',
        opacity: '0.5',
        height: '150px',
        width: '150px'
    });
});
```

http://www.w3schools.com/jquery/tryit.asp?filename=tryjquery_animation1_multicss

Animation Uses Queue Functionality

```
$("button").click(function(){var div = $("div");
    div.animate({height: '300px', opacity: '0.4'}, "slow");
    div.animate({width: '300px', opacity: '0.8'}, "slow");
    div.animate({height: '100px', opacity: '0.4'}, "slow");
    div.animate({width: '100px', opacity: '0.8'}, "slow");
});
```

http://www.w3schools.com/jquery/tryit.asp?filename=tryjquery_animation

jQuery Callback Functions

- JavaScript statements are executed line by line. However, with effects, the next line of code can be run even though the effect is not finished. This can create errors.
- To prevent this, you can create a callback function.
- A callback function is executed after the current effect is finished.
- Typical syntax: **$(*selector*).hide(*speed,callback*);**
- **Example:**

```
$("button").click(function(){
    $("p").hide("slow", function(){
        alert("The paragraph is now hidden");
    });
});
```

http://www.w3schools.com/jquery/tryit.asp?filename=tryjquery_hide_callback