# WEEK 6 – JQUERY

**GENERAL ASSEMBLY**

# FEWD



Use jQuery...

Now you're a Front End Developer.

Joe Bliss
Serial Jingle Memorizer
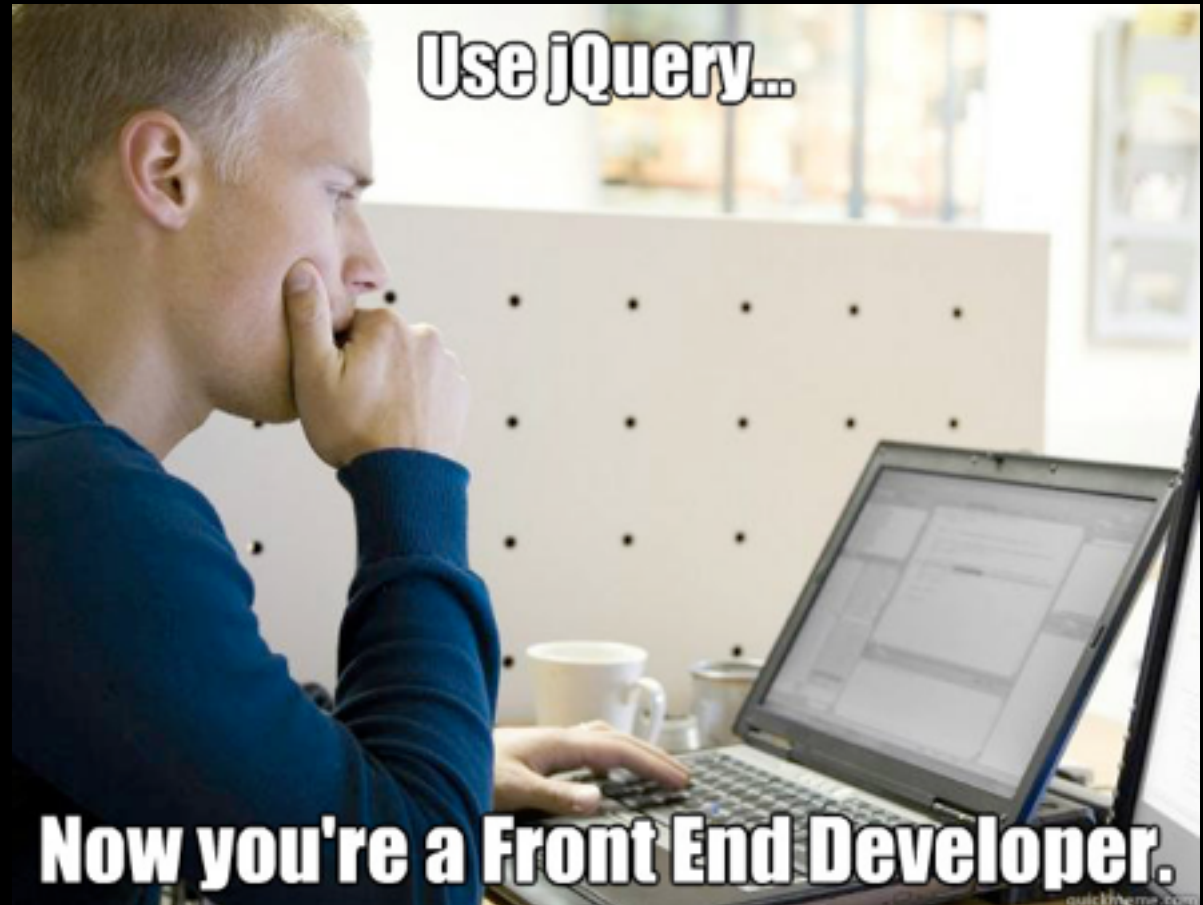
# EXERCISE – JQUERY EXERCISE

Follow the prompts in the exercise to try your hand at some jQuery.

# AGENDA

Online Dating / Calculator Box
jQuery
      HTML / CSS Manipulation
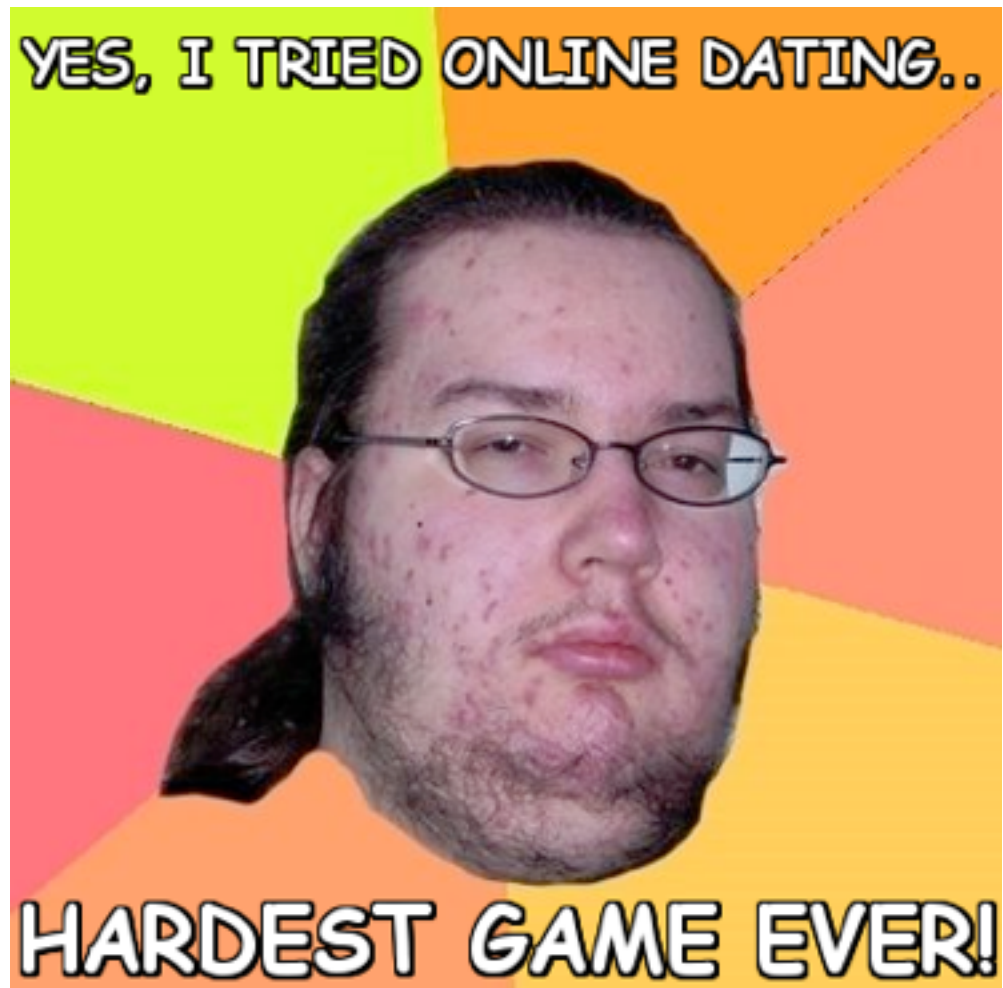      DOM Traversal
      Events
      Effects
CSS Positioning
Animation

# ONLINE DATING SITE

# ONLINE DATING SITE

# CALCULATOR BOX

Review, Questions?

# JQUERY

# JQUERY – MANIPULATE HTML

.html(htmlString)

‣ inserts (and overwrites!) the html inside the selected elements with the htmlString

.html()

‣ With no argument, it's returns the html inside the matched element

.append(string)

‣ Insert content, specified by the argument, to the end of each element in the set of matched elements.

# JQUERY – MANIPULATE CSS

.css(propertyName, rule);

‣ Changes the inline CSS values for all matched elements.

.css({prop1: val1, prop2: val2});

‣ Edit multiple properties at once.

.css(propertyName);

‣ Similar to .html(), when second arguments is not given, it returns the current value of the css property in question.

# JQUERY – HIDE AND SHOW

.hide()

‣ Hides all matched elements by setting their inline style to display: none;

.show()

‣ Reveals all matched elements by setting their inline style to display: block. If the element was originally inline, than display: inline will be used.

See also:

.hide("slow");

.show("fast");

.fadeIn(400);

.fadeOut(400);

# .CLICK()

```
.click(someFunction);
function someFunction() {
      //Execute Code in Here on Click
}
```

- or with an anonymous function -

```
.click(function() {
      //Execute Code in Here on Click
});
```

# WHAT ELSE CAN JQUERY DO?

HTML / CSS Manipulation

Effects

Document Traversal

Animation

~~Events~~

# HTML / CSS MANIPULATION

These are all ways we can directly change elements on our page.

We have already seen:

.html() and .css()

http://api.jquery.com/category/manipulation/

# GETTING / SETTING ATTRIBUTES

.attr("attribute-name")

‣ Gets the value of the matched attribute "attribute-name" of the matched element fetched by jQuery

.attr("attribute-name", "new-value")

‣ Changes the value of the matched attribute "attribute-name" of the matched element fetched by jQuery to "new-value"

# GETTING ATTRIBUTES

```
<a href="http://www.google.com">Google</a>
<img id="logo" src="smiley.jpg">
```

```
$("a").attr("href"); //Returns "http://www.google.com"
$("#logo").attr("src"); // Returns "smiley.jpg";
$("#logo").attr("id"); // Returns "logo";
```

# SETTING ATTRIBUTES

<img id="puppies" src="images/puppies.png">

Calling:

$("#puppies").attr("src", "images/new-puppies.jpg");

Will change the image on the page:

<img id="puppies" src="images/new-puppies.jpg">

# CODEALONG – PUPPY SWAP

http://codepen.io/josephjbliss/pen/dIAxt

# EXERCISE – JQUERY CITY

User clicks on thumbnail.

- the larger image is replaced by the corresponding thumbnail image.

Hint: use .attr() ...

# HTML INSERTION

.append("content")

‣ Insert "content" at the end of each matched element.

.prepend("content")

‣ Insert "content" to the beginning of each matched element

.before("content")

‣ Insert "content" before all matched elements

.after("content")

‣ Insert "content" after all matched elements

# GETTING / SETTING FORM VALUES

.val()

‣ Gets the text in the input and returns it as a String

.val("Some string")

‣ Set the text in the input to "Some string"

<input name="age" type= "text">

$('[name="age"]').val(); // Returns the value stored in the input

$('[name="age"]').val("21"); // Sets the text in the input to "21"

# CODEALONG – HTML INSERTION

.append(), .prepend(), .after(), .before(), .val() in action:

http://codepen.io/josephjbliss/pen/yJoib

# EXERCISE – LIST MAKER

Create a Grocery List by adding additional <li> items to the <ul>. User enters item into input, clicks "Add", text is scraped from input, and added to an <li> at the end of the <ul>.

Cue cheesy stock photo ->

# MANIPULATING CSS CLASS NAMES

$("div").addClass("special");
    //adds class="special" to all <div> elements

$("div").removeClass("special");
    // removes class="special" from all <div> elements

# MANIPULATING CSS CLASS NAMES

```
$("div").toggleClass("special");
// toggles the class="special" on all <div> elements (a
dds it if it doesn't exist, and removes it if it does)


if ($("#myElement").hasClass("special")) {
    // do something here
}
```

# WHAT DOES THIS DO?

```
<span>Click</span> <span>on</span>
<span>me</span> <span>to</span>
<span>change</span> <span>color.</span>


$("span").click(function(){
    $("span").css("color", "red");
});
```

# WHAT DOES $(THIS) DO?

When you are in an event (for instance a .click()), calling $(this) will return the object that initiated that event.

```
$("span").click(function(){
    $(this).css("color", "red");
});
```

Incorporate this change here:

http://codepen.io/josephjbliss/pen/qyrfj?editors=101

As we click, each word will become red, rather than ALL the words turning red at once.

# CODE REFACTORING

Code refactoring is the process of restructuring existing computer code without changing its external behavior.

http://en.wikipedia.org/wiki/Code_refactoring

# CODEALONG – COLOR SWITCHER

Refactor the brute-force approach to the Color Switcher.

http://codepen.io/josephjbliss/pen/qvCau

# EXERCISE – REFACTOR–ATOR BOX

Discuss with your neighbor how you might think about restructuring your code.

You can change the HTML, CSS, Javascript/jQuery. The end result should look and act EXACTLY the same. In how few lines of code can you do the same task?

# EFFECTS

"These include simple, standard animations that are frequently used."

http://api.jquery.com/category/effects/

# HIDE / SHOW / TOGGLE

.hide()
- ‣ Hide the matched elements

.show()
- ‣ Show the matched elements

toggle()
- ‣ If object is currently hidden, show()
- ‣ If object is currently shown, hide()

# FADEIN / FADEOUT

.fadeIn(400);

‣ Fade the matched elements in over 400 ms

.fadeOut(400);

‣ Fade the matched elements out over 400 ms

# CODEALONG – SIMPLE SLIDESHOW

Let's do this via brute force approach first.

# SLIDEUP / SLIDEDOWN

.slideUp(500);

- ‣ Hide the the matched elements with a sliding-up animation over 500 ms

.slideDown(500);

- ‣ Show the the matched elements with a sliding-up animation over 500 ms

.slideToggle(500);

- ‣ If shown, slideUp(500)
- ‣ If hidden, slideDown(500)

# CHAINING COMMANDS

Chain chain chain … Chain of fools …

Perform multiple effects one after the other. Only works when called on the SAME object.

$("#p1").fadeIn(800).slideUp(2000).slideDown(2000);

When you chain commands, effects are queued. Normal commands execute immediately after one another.

# CALLBACK FUNCTION

A callback function is a function that runs AFTER an effect is completed. It is super useful for timing things around animations.

For example:

```
$("p").fadeOut(2000 ,function(){
    alert("The paragraphs are now gone.");
});
```

// This will fadeOut all the paragraphs on the page over 2 seconds, then, once that has completed, send an alert.

# ANIMATIONS – .ANIMATE();

```
$("#myElement").animate({
    "opacity": ".3",
    "width": "500px",
    "height": "700px"
  }, 2000);
```

The .animate() method allows us to create animation effects on (most) numeric CSS properties.

# .ANIMATE() THESE THINGS:

"border-width"

"border-bottom-width"

"border-left-width"

"border-right-width"

"border-top-width"

"margin"

"margin-bottom"

"margin-left"

"margin-right"

"margin-top"

"padding"

"padding-bottom"

"padding-left"

"padding-right"

"padding-top"

"height"

"width"

"font-size"

"top"

"left"

"bottom"

"right"

# DOCUMENT TRAVERSAL

How we "get-around" within our document. How each element is related to all the other elements on the page.

http://api.jquery.com/category/traversing/

# IT'S ALL IN THE FAMILY

.children()
- ‣ Get the child elements

.parent()
- ‣ Get the parent element

.siblings()
- ‣ Get the sibling elements

# BROTHERS AND SISTERS

.next()

‣ Get the immediately following sibling element

.next("some selector")

‣ Get the immediately following sibling element but only if it matches "some selector"

.prev()

‣ Get the immediately preceding sibling element

# BROTHERS AND SISTERS

.index()

‣ Returns an item's index within its siblings.

.eq(someNumber)

‣ Returns the item at the specified index (someNumber)

# CODEALONG – ACCORDION

Let's implement a jQuery accordion using sliding effects and DOM traversal functions.

# CODEALONG – LESS SIMPLE SLIDESHOW

We want to make this generic for ANY number of slides and any navigation? How can we do this?

Think about the DOM tree structure ...

# CSS POSITIONING

Static
Fixed
Relative
Absolute

What's your favorite posish?

# POSITION: STATIC;

Everything that we've seen so far has been "position: static;" by default. This is the default for all elements.

You cannot set right, left, top, bottom values to elements with position: static;

A static element is said to be "not positioned" and an element with its position set to anything else is said to be "positioned".

# POSITION: FIXED;

Fixed-position elements don't move when the browser scrolls. They are placed relative to the browser window (top, left, bottom, right). Elements with position: fixed; are removed from the normal flow of the page.

Usage: Persistent navigation, "Modal" divs

# POSITION: ABSOLUTE;

An absolutely-positioned element is positioned relative to it's nearest parent with a position other than static (relative, absolute, fixed). Often, this is the browser window.

Absolutely-position elements are taken out of the normal flow of the page.

Usage: Animation!, Placing an element within a parent.

# EXERCISE – FARM PARTY

Set clouds_2.gif as the background image.

Create a div for the ground/grass with green background and appropriate height - position it at the bottom with position: fixed; or position: absolute;

Create an h1 and display FARM PARTY!. For a bonus, use a fun custom font (like from Google Web Fonts).

For each of the characters you want to animate, create an <img> for it on the page.

Position each of the <img> elements with position: absolute; or position: fixed;

Add animations to each of the <img> elements.