WEEK 7 - JQUERY AND RESPONSIVE DESIGN

GENERAL ASSEMBLY

FEWD

Joe Bliss

Wishes he was on a beach somewhere ...

AGENDA

Share Online Dating Sites

Questions on Calculator Box

Refactor Calculator Box

jQuery

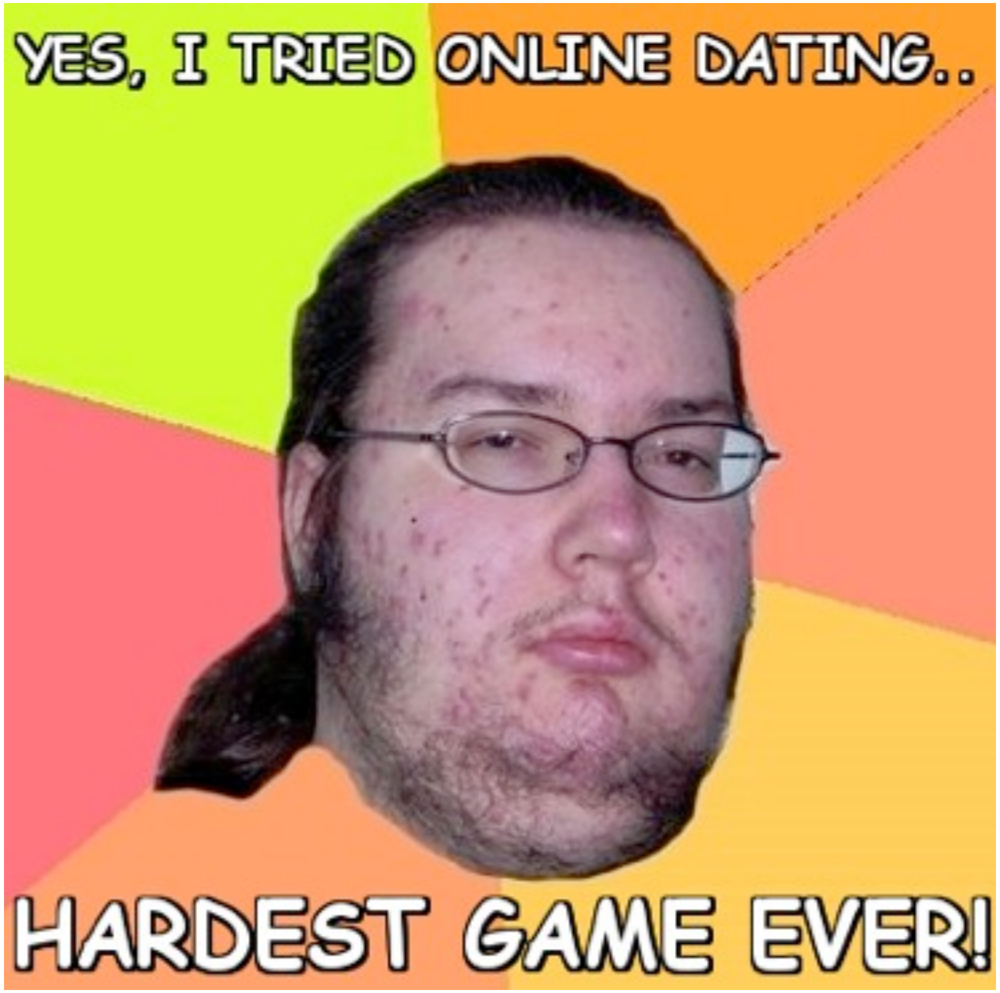
jQuery Animation

Responsive Design

ONLINE DATING SITE

You look different in your online profile.
I was impressing you with my Photoshop skills.
35

ONLINE DATING SITE



CALCULATOR BOX

Review the assignment.

How can we make it better?

CODE REFACTORING

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

(source: http://en.wikipedia.org/wiki/Code_refactoring)

CODEALONG - REFACTOR CALCULATOR

JQUERY RE-INTRO

jQuery IS Javascript

Or more accurately, jQuery is a cross-browser Javascript library designed to simplify the client-side scripting of HTML applications.

WHY DO WE USE IT?

The official jQuery slogan: "Write less, do more."

Different browsers handle DOM manipulation, transparency effects, and animation in different ways. jQuery abstracts these out so we only need to write the bare minimum. Here's a simple example:

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

- JS

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

- jQuery

THE $(“X”) FUNCTION - GET ALL THE “X”

$(“x”);

- Query the document and return all the “x” elements.
- This could also be written as: jquery(“x”); We use the $ as a convention to streamline.

$("li a");

- Will return all of the <a>’s that are within a <li>

Any CSS selector can go here (as well as some jQuery-specific selectors that are built-in).

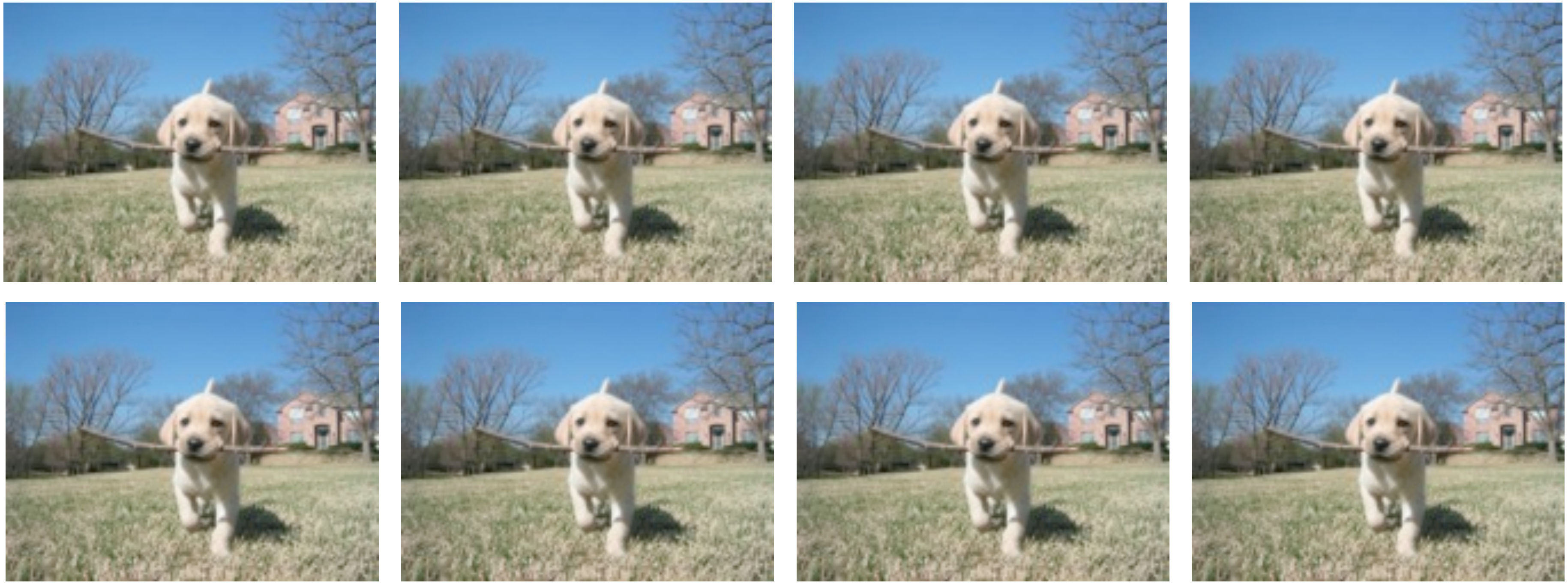
$(“#TENNISBALL”)

Go to the HTML and bring me back the object with the id=“tennisball”.

jQuery is playing fetch.

$(“.TENNISBALL”)

Go to the HTML and bring me back ALL the objects with the class=“tennisball”.



JQUERY COMMANDS

```
$("li").hide( );
$(“li”).html(“New Item”);
$(“li”).css(“color”, “red”);
$("li").fadeIn( );
```

SELECTING ELEMENTS

```
$("div"); // selects all HTML div elements

$("#myElement"); // selects one HTML element with ID "myElement"

$(".myClass"); // selects HTML elements with class "myClass"

$("p#myElement"); // selects paragraph elements with ID "myElement"

$("ul li a.navigation");  // selects anchors with class "navigation" that are nested in list items
```

SELECTING ELEMENTS

```
$("p > a"); // selects anchors that are direct children of paragraphs

$("input[type=text]"); // selects inputs that have specified type

$("a:first"); // selects the first anchor on the page

$("p:odd"); // selects all odd numbered paragraphs

$("li:first-child"); // every list item that's first child in a list
```

HOW TO INCLUDE JQUERY

Option 1: Download and store locally:

- Go to http://jquery.com/ and click download button
- Store file in js folder.
- Add script tag to HTML like any other script

Option 2: Include from Google API or other CDN:

- <script type="text/javascript" src="http://ajax.googleapis.com/ajax/libs/jquery/1.10.2/jquery.min.js"></script>

Which is better?

http://encosia.com/3-reasons-why-you-should-let-google-host-jquery-for-you/

WHERE DO WE PUT JQUERY?

What happens if we put our jQuery in the <head>?

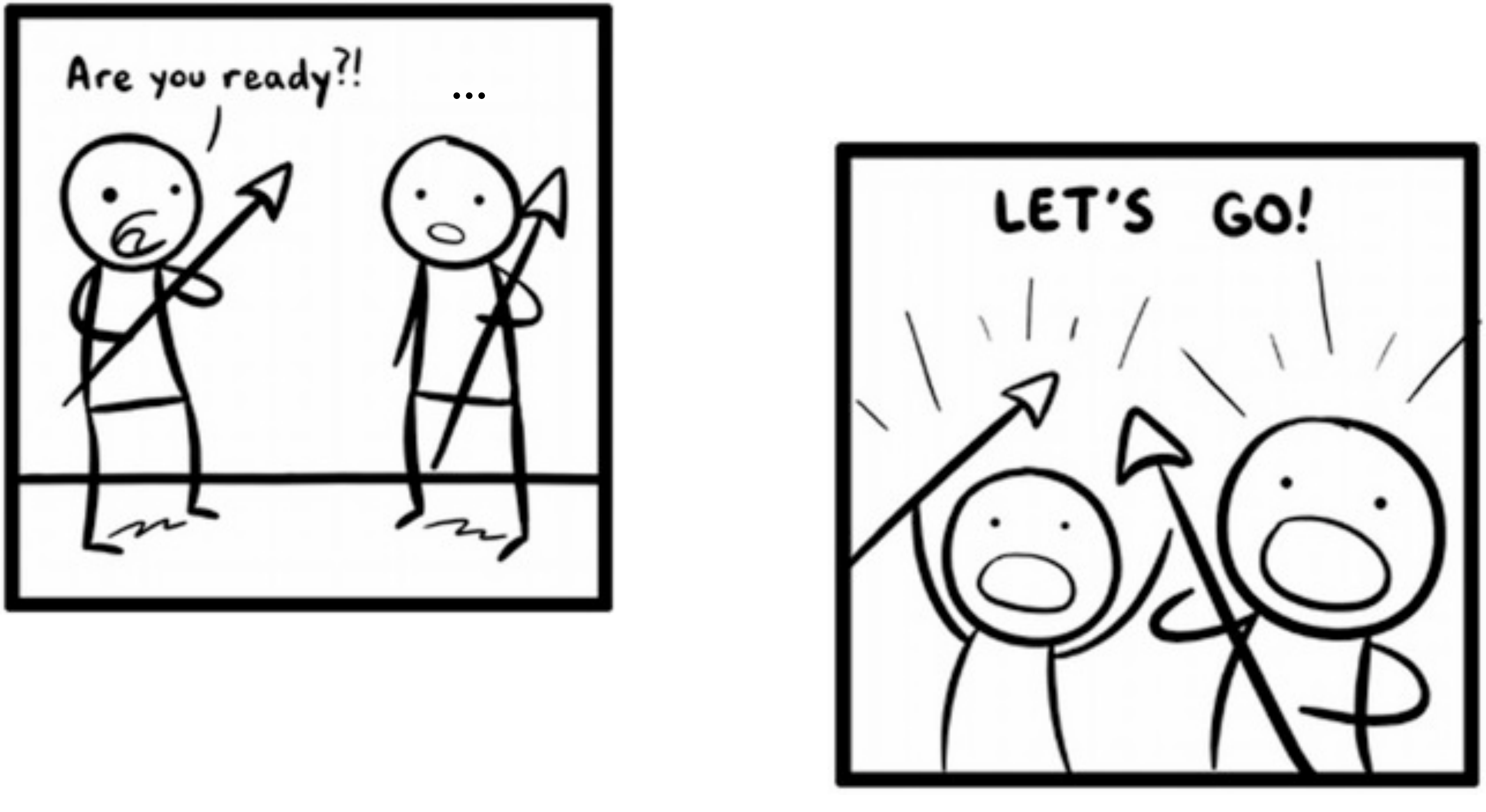
Zebra-stripes

RUNNING CODE WHEN THE DOM IS READY

```
$(document).ready(function(){
  $("li.blue").css(“background-color”, “blue”);
});
```

Using $(document).ready(), we are able to execute code only once the document is done rendering (i.e. the DOM is ready).

RUNNING CODE WHEN THE DOM IS READY



ANONYMOUS FUNCTIONS

In the previous example, we used an anonymous function. Some commands expect a function as an argument. The anonymous function lets us use a function without naming and separately defining it.

MANIPULATING CSS CLASS NAMES

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

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

$("div").toggleClass("content");
// toggles the class "content" on all <div> elements (adds it if it doesn't exist, and removes it if it does)

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

MANIPULATING CSS STYLES

```
$("p").css("width", "400px");
// adds a width to all paragraphs

$("#myElement").css("color", "blue")
// makes text color blue on element #myElement

$("ul").css("border", "solid 1px #ccc")
// adds a border to all lists
```

MANIPULATING CONTENT

```
$("#myElement").html("<p>This is the new content.</p>");
 // content inside #myElement will be replaced with that specified

$("#myElement").text("This is the new content.");
// text content will be replaced with that specified

$("#myElement").append("<p>This is the new content.</p>");
// keeps content intact, and adds the new content to the end
```

EVENTS - THE CLICK

```
$("a").click(function() {
    // do something here when any anchor is clicked
    // Uses anonymous function, as before.
});
```

OTHER MOUSE EVENTS

.mousedown()

The mouse has been pressed.

.mouseup()

The mouse has been released.

.mouseenter()

The mouse has entered an object.

.mouseleave()

The mouse has left an object.

ANIMATIONS - HIDE AND GO SHOW

```
$("#myElement").hide(1000);

$("#myElement").show(1000);

$("#myElement").toggle(1000);
//If hidden, show. If shown, hide.
```

ANIMATIONS - FADING ... FADING ...

```
$("#myElement").fadeOut(1000);
//Slowly animate opacity to 0, then display: none;

$("#myElement").fadeIn(1000);
//Slowly animate opacity to 1, then display: block;
```

CODEALONG - JQUERY CITY

CODEALONG - SIMPLE SLIDESHOW

FORM EVENTS

```
$(“#input1”).focus(function() {
 //User has placed his cursor on an element or the element is selected
});

$(“#input1”).blur(function() {
 //The element has lost focus (eg. user has clicked somewhere else)
});

$(“#input1”).submit(function() {
 //Form has been submitted (user clicked “submit” / pressed return)
});
```

EXERCISE - LIST MAKER

With a partner, create the grocery list app.

Create a Grocery List by adding additional <li> items to the <ul>.

You will use click(), val(), and append()

- Google "jquery append" to see if you can figure this out ...

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:

borderWidth

borderBottomWidth

borderLeftWidth

borderRightWidth

borderTopWidth

margin

marginBottom

marginLeft

marginRight

marginTop

padding

paddingBottom

paddingLeft

paddingRight

paddingTop

height

width

fontSize

bottom

left

right

top

CHAINING COMMANDS

Chain chain chain ... Chain of fools ...

```
$("#p1").css("color","red").slideUp(2000).slideDown(2000);
```

CALLBACK FUNCTION

A callback function is a function that runs AFTER another effect is completed.

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.
```

RESPONSIVE DESIGN

Designing for the optimal viewing experience agnostic of the device on which the site is being viewed.

We are going to learn how to BUILD responsive sites.

“One site for every screen.”

RESPONSIVE DESIGN

Let’s explore some responsive sites:

http://thenextweb.com/

http://www.abookapart.com/

http://trentwalton.com/

http://www.time.com/time/

What is happening that makes it “Responsive”?

RESPONSIVE DESIGN

Actually is a combination of the following:

Fluid Layout - Use of %'s

Media Queries

- Resizing Elements
- Floating / Unfloating (i.e. rearranging columns)
- Removing / Showing Content

Use of em's

FLUID LAYOUT

Startup Matchmaker

What’s the problem with ONLY fluid layout?

MEDIA QUERIES

The first media query we learn has nothing to do with responsive design. The first media query we will learn is the print media query.

Can be either within our stylesheet:

```
@media print {
    /* Styles go here */
}
```

Or linked as its own stylesheet:

```
<link rel="stylesheet" media="print" type="text/css" href="css/print.css">
```

MEDIA QUERIES

Link to separate stylesheet:

```
<link rel=”stylesheet” media=”screen and (min-width: 768px)”
href=”css/768.css”>
```

Within stylesheet:

```
@media screen and (min-width: 768px) {
/* Styles go here */
}
```

Both assign styles for browser widths wider than 768px.

MEDIA QUERIES

Link to separate stylesheet:

```
<link rel=”stylesheet” media=“only screen and (min-width: 768px)”
href=”css/768.css”>
```

Within stylesheet:

```
@media only screen and (min-width: 768px) {
/* Styles go here */
}
```

Both assign styles for browser widths wider than 768px.

MEDIA QUERIES

Classic “Break Points” to target:
960px, 768px, 480px, 320px

Can be multiple:

```
@media only screen and (min-width: 320px) and (max-width: 480px) {
    /* These styles will only apply on devices between 320 and 480
    pixels wide. */
}
```

CODEALONG - ROXY CAT

Let’s add some media queries to Roxy!

EM ... WHAT?

“Equal to the computed value of the ‘font-size’ property of the element on which it is used.”

```
.element {
   font-size: 20px;
   line-height: 1.2em; //i.e. 24px
   margin: 2em 0; //i.e. 40px 0
}
```

CARPE THE EM!

Set a font-size on the <body> using pixels.

Set your typography based on ems.

Then, in your media queries at smaller sizes, set that <body> font size to be smaller. The other values will cascade down.

I EM WHAT I EM

RESPONSIVE RULES OF THUMB

Use pixels for font size on <body> and other top-level elements.

Use percents for widths, margins, block padding.

Use em's for typography (font-size, line-height, etc.)

Do this:

<meta name="viewport" content="width=device-width, initial-scale=1.0">

FURTHER READING

http://johnpolacek.github.io/scrolldeck.js/decks/responsive/

TESTING MEDIA QUERIES IN BROWSER

http://www.jamus.co.uk/demos/rwd-demonstrations/

FINAL PROJECT MILESTONE 2

The next step in your Final Project, which will be due in Week 8, is to provide:

- a DRAFT of HTML/CSS of one page of your site
- Pseudocode of your JS interactions

Final Projects will be due on the last day of class. They will also be presented that day.