Refactor

Lesson 12

FEWD
GENERAL ASSEMBLY

Learning Objectives

- Understand when to use arrays and for loops
- Define refactoring and describe why it is important.
- Learn the basics of CSS/JS refactoring and be able to apply these concepts to your own code
- Describe the concept of "this" as it applies within jQuery anonymous functions
- Know the different ways to debug code and how to apply the concepts to your own code

Agenda

- Functions Review
- Arrays and For Loops
- Refactoring
- "This" keyword
- Debugging
- Lab Time

Functions Review

Functions Review

Open: **Functions Review Codepen**

JavaScript Arrays

What is an Array?

An **array** is a special variable that can hold more than one value at a time.

Traditional Variables

Typically, you'd create a list of cars like so:

```
var car1 = "Saab";
var car2 = "Volvo";
var car3 = "BMW";
```

Array Syntax

Arrays are simply sets of data surrounded by square brackets [], and separated by commas:

Creating Arrays

The most common way to create an array is the same as creating any other variable...

Except, this time, you assign the variable a list of objects:

```
var cars = ['Saab','Volvo','BMW'];
```

Array Data

An array can contain the following:

- strings
- numbers
- boolean
- other arrays
- and more...

Array Examples

```
var compOptions = ["rock", "paper", "scissors"];

var randomNum = [23, 13, 28, 5, -30];

var mixedData = [true, "Hello World", ["Cookies", 45], 13];
```

Accessing Arrays

To actually access the data in an array, you will simply access the variable + the index of the desired element wrapped in square brackets:

```
varName[0];
```

Accessing Arrays Example

```
// Create the array
var cars = ['Saab','Volvo','BMW'];

// Get the 'BMW' value from the array
cars[2];
```

IMPORTANT!!

Array indexes are 0-based!

Meaning, array indexes **start with 0**.

```
var cars = ['Saab','Volvo','BMW'];
              ↑       ↑      ↑
              0       1      2
```

Array Props & Methods

The real strength of JavaScript arrays are the built-in array properties and methods:

```
myArray.length;

myArray.toString();

myArray.pop();

myArray.push();

myArray.shift();
```

Code Along

Open: JS Arrays Codepen

Pulse Check

Key Objective:

Create an array that contains all of the students in the class. Then practice accessing/updating the array via its properties and methods.

Timing:

- **10 minutes** - As a class, create a student array and access the data inside of it

JavaScript Loops

What is a Loop?

A **loop** is away to run the same code block a certain number of times.

There are many different types of loops, but they all essentially do the same thing (with slight variations)...

Today, we will focus on the **for loop**

For Loops

A **for loop** repeats until a specified condition evaluates to false:

```
for (var i=0; i < 10; i++) {
    // Some code to be repeated...
}
```

For Loop Syntax 1

The first statement sets a variable before the loop starts:

```
for (var i=0; i < 10; i++) {
    // Some code to be repeated...
}
```

For Loop Syntax 2

The second statement defines the condition for the loop to run:

```
for (var i=0; i < 10; i++) {
    // Some code to be repeated...
}
```

For Loop Syntax 3

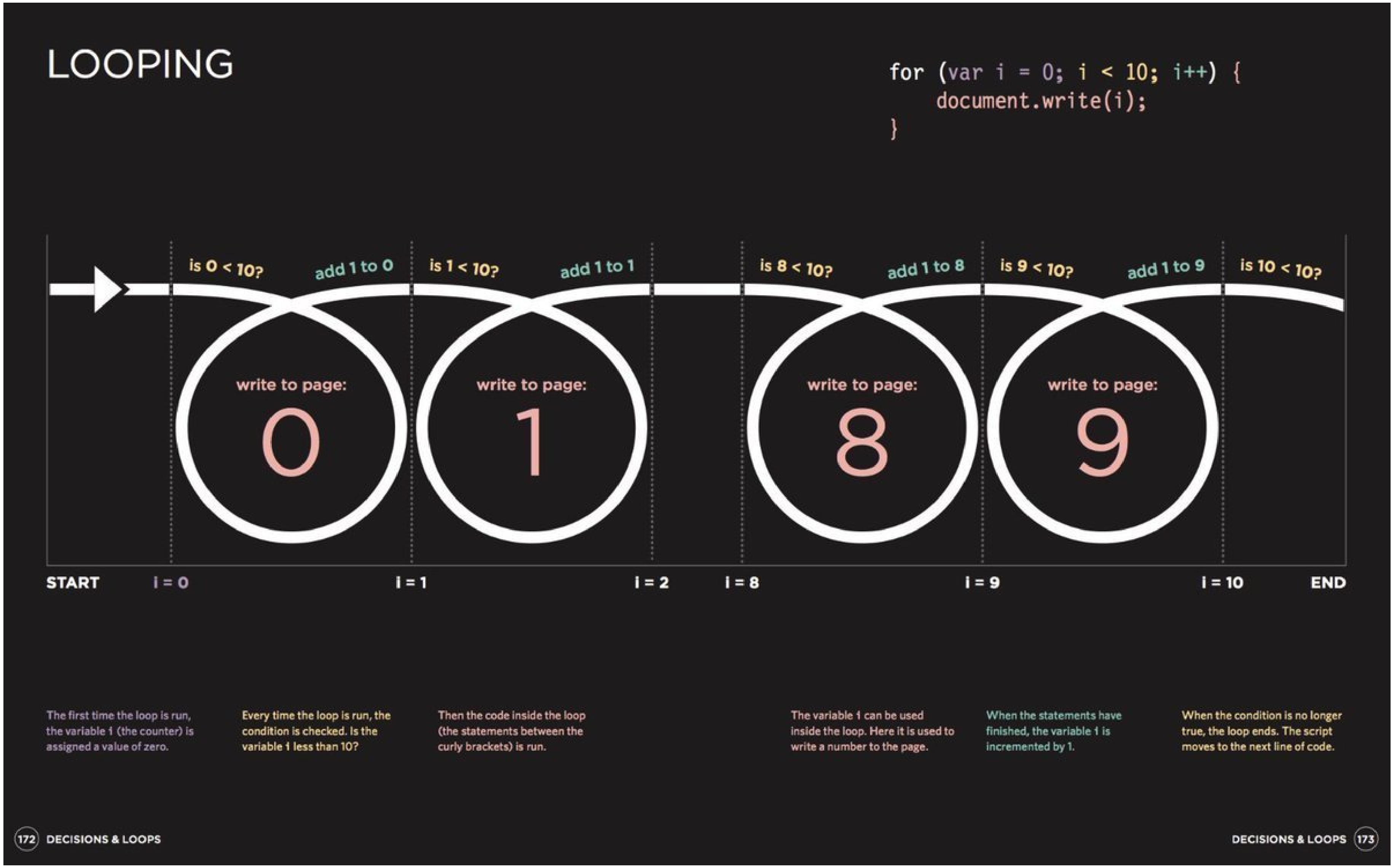
The last statement increments the counter by one each time the code block in the loop has been executed.

```
for (var i=0; i < 10; i++) {

    // Some code to be repeated...

}
```

For Loop Illustration



LOOPING

```
for (var i = 0; i < 10; i++) {
    document.write(i);
}
```

The first time the loop is run, the variable i (the counter) is assigned a value of zero.

Every time the loop is run, the condition is checked. Is the variable i less than 10?

Then the code inside the loop (the statements between the curly brackets) is run.

The variable i can be used inside the loop. Here it is used to write a number to the page.

When the statements have finished, the variable i is incremented by 1.

When the condition is no longer true, the loop ends. The script moves to the next line of code.

172 DECISIONS & LOOPS

DECISIONS & LOOPS 173

For Loop Example

Let's type this in our Console and see what we get...

```
// Basic For Loop
for(i=0; i<10; i++){
    console.log('This is iteration number: ' + i);
}
```

For Loop + Array

For loops are an extremely useful way to loop over an array, and access each item in that array.

For Loop + Array Example

```
var cars = ['Saab','Volvo','BMW'];

for(i=0; i<cars.length; i++){
        console.log("Car index: " + i + " and Car name: " + cars[i]);
}
```

Pulse Check

Key Objective:

Revisit the **JS Arrays Codepen**. Refactor #2 to use the for loop syntax to access each student in the array and display it in the HTML.

Timing:

- **10 minutes** - As a class, refactor accessing the array using for loops.

Refactoring

What is Refactoring?

Refactoring is the process of making code more efficient without changing its functionality.

What it isn't

An exact science...

Why Refactor?

- To reduce or eliminate redundancy
- Make code easier to read
- Make code more maintainable
- And more...

CSS Refactoring

- Remove inline styling
- Replace repeated styles with classes
- Rename classes/ids for readability
- Organize CSS
  - Group by section
  - Order by precedence (tag selectors at top, id selectors at bottom)
  - Create classes for large CSS changes in JS
  - Remove unnecessary CSS

JS Refactoring

- Use functions
- Use variables
- Use arrays
- Combine jQuery selectors
- Combine jQuery property changes into objects
  - .css, .attr, etc
- Chain jQuery function calls

Code Along

Open: **Refactoring CodePen**

Pulse Check

Key Objective:

Refactor the HTML, CSS, and JS in the Codepen to be as efficient as possible. Be sure to use DRY (Don't Repeat Yourself) principles wherever possible.

Timing:

- **10 minutes** - As a class, examine the HTML, CSS, and JS in the Codepen and refactor it.

Tips:

- Don't Repeat Yourself!
- Be on the lookout for any extraneous code in your HTML and CSS
- Remove it if it is not necessary

"This" Keyword

What is "This"?

"This"

In jQuery, the **this keyword** refers to the selected object.

"This" Example

```
$("p").on("click",function(){
    $(this).fadeOut(500);
});
```

Assuming there are multiple <p> elements on the page, the above code will **only** affect the <p> element that was actually clicked.

Code Along

Open: **jQuery This CodePen**

Pulse Check

Key Objective:

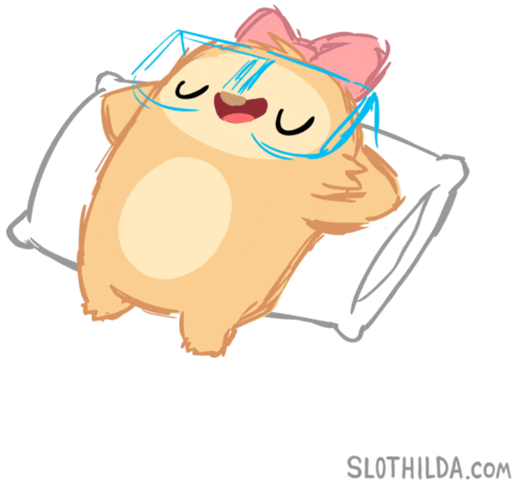
Using the "this" keyword in jQuery, refactor the JS in the Codepen to be as efficient as possible.

Timing:

- **10 minutes** - As a class, use the "this" keyword to refactor the jQuery.

Tips:

- Don't Repeat Yourself!
- Be on the lookout for any extraneous code in your HTML and CSS
- Remove it if it is not necessary



Break time!

Let's take 5-10 minutes to decompress...

Debugging

What is Debugging?

Debugging is the act of systematically going through your code to find/fix an error.

Start Debugging

Always start by defining the problem...

- The image is not displaying
- My form is not submitting
- None of my code works
- And so on...

More Debugging...

This will tell you where to start hunting for your problem:

- Image not displaying...
  - find the code that should make it show up
- Form not submitting...
  - Are you listening for a .submit() method?
- None of my code works...
  - Is your JS file linked properly in your HTML?
  - Are you seeing a syntax error? What line?
  - Check console...

Debugging: Level 1

Check for errors (red text, aligned right) in console

To access debugging console in Chrome:

PC: CTRL + SHIFT + J

Mac: COMMAND + OPTION + J

Click the error

The location may not be correct, but it is a good place to start.

Debugging: Level 2

So, no red errors in the console, but still not getting the desired result?

Try placing a console.log() at different places in your code.

```
var stringOfNames = "";

["Bob","Joe"].forEach(function(element){
    stringOfNames -= element + ",";
    console.log(stringOfNames);
});
```

Debugging: Level 3

- Use the debugger in Chrome:
  - Set a breakpoint
  - Run the code
  - Step through the code until you get to the error
  - Variable values display on the right
  - You can switch to the console to run code or check value of variable

Debugging: Level 4

Get Help!

- Try a search using your preferred search engine (Google)
- Be ready to clearly articulate the problem (be sure to append the coding language to your query)
- If all else fails, ask your instructor

Debugging Resources

JavaScript Debugging Resources

Code Along

Open: **starter_code/debug**

Pulse Check

Key Objective:

Open the index.html file under the **starter_code/debug** folder. Debug the file according to the directions in the *debug.js* file

Timing:

- **10 minutes** - As a class, debug the JS file so that it works properly

Tips:

- Follow the debug steps listed in the previous slides.
- Make copious use of your console AND console.log statements whenever necessary.

Lab Time

Open: **lesson_12/starter_code/color-scheme**

Directions:

1. Open the **lesson_12/starter_code/color-scheme** folder
2. Using the "this" keyword, refactor the index.js file to be more efficient

Timing:

- **40 minutes** - refactor the JS using the "this" keyword in jQuery
- **10 minutes** - students share their work
- **5 minutes** - Review instructor solution file together

Tips:

- Take it one step at a time!
- Use the previous "This" CodePen for inspiration
- It can be done **without** editing the HTML/CSS
  - But, feel free to edit it if you must

Exit Tickets

Take 5-10 minutes to give us some

(Link is in Slack Room)

Learning Objectives Review

- We understand when to use arrays and for loops.
- We defined refactoring and described why it is important.
- We learned the basics of CSS/JS refactoring and are able to apply these concepts to our own code.
- We described the concept of "this" as it applies within jQuery anonymous functions.
- We know the different ways to debug code and how to apply the concepts to our own code.

Week 6 Homework

- Assignment: **Citipix (Refactored)**
- Due: **Wednesday, June 12th at 11:59pm ET**
- Giving everyone an extra couple days to finish
- Remember to commit/push your changes to your fork when you are done.
- Grading rubric can be found in the **Assignment** folder
- **FOLLOW THE RUBRIC!**

Next Class...

Lesson 13 - Responsive Basics