FEWD - WEEK 6

WILL MYERS

Freelance Front End Developer

SLIDES

http://www.slideshare.net/wilkom/fewd-week6-slides

YOUR WEEKLY FEWD GITHUB REPOSITORY

- Use the '+' button in the top-left of GitHub Desktop (*Create* tab)
- Create a new repository called *'FEWD_Week6'*
- Choose the [home]/FEWD folder for the local path
- Open this repo folder in your editor
- Commit the changes and publish the *FEWD_Week6* repository to github.com

YOUR WEEKLY WORKING FILES FROM ME

To get the *week6_working_files* you should just be able to select the *ga-fewd-files* repository in GitHub Desktop and press 'Sync'. This should pull in this weeks folder from github.com.

If you any difficulties you should just re-clone the *ga-fewd-files* repository.

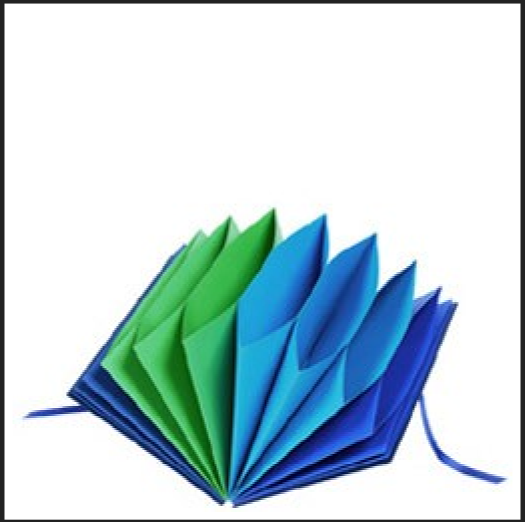
Copy the whole *week6_working_files* into your *FEWD_Week6* repository and commit and publish to github.com

REVIEW OF LAST WEEK'S ASSIGNMENT

AGENDA

- Collection Of Data
- Manipulating Collections

ARRAYS COLLECTIONS



ARRAYS

What if we had a collection of images that we wanted to display to the screen one at a time?

How could we store all the images?

ARRAYS

An array is a list **object** with **built in methods** for things like:

- adding to the list
- removing from the list
- traversal of the list.

DECLARING ARRAYS

```
var myArr = new Array();
```

- declaring an empty array using the Array constructor.

DECLARING ARRAYS

```
var myArr = [ ];
```

- declaring an empty array using literal notation.

DECLARING ARRAYS

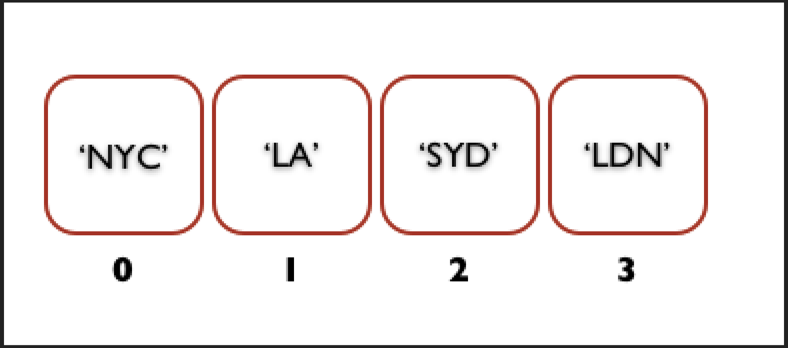
```
myArr = ['Hello', 54.3, true];
```

- Arrays are filled with elements: i.e. myArr3 = [element, anotherElement];
- Elements can contain strings, numbers, booleans, and more.

DECLARING ARRAYS

If you leave a blank spot in an array it creates a blank shelf space (undefined) placeholder.

ARRAYS INDEXING



ARRAYS INDEXING

Array elements can be fetched by their index number (starts from 0).

```
myArr = ['Hello', , 54.3, true];

console.log(myArr[0]); //prints Hello
console.log(myArr[1]); //prints undefined
console.log(myArr[2]); //prints 54.3
console.log(myArr[3]); //prints true
```

ARRAYS INDEXING

We can insert new values into any space in the array using the positions index.

```
myArr[1] = 'Stuff';
```

ARRAYS INDEXING

We can overwrite all the elements of an array simply by giving the array new values or by setting an array equal to a different array.

```
var fruits = ['Apples', 'Oranges', 'Pears', 'Bananas'];
var myArr=[1,2,3];
myArr = fruits;

console.log(myArr); //prints Apples, Oranges, Pears, Bananas
```

ARRAY LENGTH

What if I would like to know how long my array is (how many elements)?

```
console.log(myArr.length); //prints 4
```

ARRAY METHODS

The Array object has many built in methods for doing stuff with arrays. Here are two common methods:

`Array.push()` *adds an item to the end of an array*

```
var myArr = [1,2,3];
myArr.push(4); //myArr === [1,2,3,4]
```

`Array.pop()` *removes an item from the end of an array*

```
var myArr = [1,2,3,4];
var popped = myArr.pop(); //myArr === [1,2,3]; popped = 4;
```

ARRAYS EXERCISE

Open *week5_working_files/arrays_exercise*

ITERATE OVER ARRAY

- Computers can repeatedly execute lines of code very quickly (in milliseconds and nanoseconds)
- Combined with conditions (if) computers can process large quantities of data quickly and make "intelligent" decisions based on this data.
- Sequentially processing a list of data and doing something with the data is one of the most common activities in programming.

ITERATE OVER ARRAY - REPEAT LOOPS

`for` loop:

```
for (var i = 0; i < 5; i++) {
    //i runs from 0 to 4 in this loop.
};
```

`while` loop:

```
var n = 10;
while(n--){
    console.log('n is', n); //n runs from 9 to 0
};
```

ITERATE OVER ARRAY

The `Array.forEach` method also allows you to run code using each element from the array as a value

You pass an **anonymous function** with pre-defined arguments

```
var fruits=["Banana","Apple","Pear"]
    fruits.forEach(function(element,index){
    console.log(element, "is at position", index);
});
```

`element` is the item from the array

`index` is the item's position in the array

MORE ON ARRAYS

For many more Array methods see:
https://developer.mozilla.org/en-US/docs/JavaScript/Reference/Global_Objects/Array

CAROUSEL

Open *week5_working_files/carousel_animals*

WEATHER APPLICATION

Can you build a weather application that works in the same way as your CitiPix assignment?

- The user inputs a temperature in Celsius
- This temperature gets converted to Fahrenheit and displayed on the page
- Change the display of an image on the page if the temperature is cold or hot (< 20 degrees Celsius or >= 20 degrees Celsius)

ROCK PAPER SCISSORS

Open *week6_working_files/rock_paper_scissors*

AGENDA

- Refactor
- This Keyword
- Debugging Techniques

REFACTOR

- Making code more efficient without changing functionality.

REFACTOR

The process of rewriting code without changing functionality

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

CSS REFACTOR

- 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 REFACTOR

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

REFACTOR

Open *week6_working_files/refactor*

SWITCH BLOCKS

When you end up writing a lot `if ... else` syntax in your code (for example in the Rock Paper Scissors game) you can use a different syntax (as follows) to make your code a bit cleaner and clearer:

```
switch(someValue){
    case "value1":
        doThis();
        break;
    case "value2":
        doThat();
        break;
    default:
        doSomethingElse();
}
```

It is important have the `break;` keyword at the end of each `case` statement, so as to exit the `switch` block correctly.

SWITCH BLOCKS

Let's refactor Rock Paper Scissors with a `switch` block

KEYWORD: "THIS"

In JavaScript `this` is the keyword for a **context** object. It is used inside a function block.

You can think of the context as the **owner** of a function.

If you write a function in the **global context**, then `this` will refer to the global `window` object.

```
//written in the global window context of your JavaScript file
function doSomething(){
    return this; //this refers to the global window object
}
console.log (doSomething() === window);// returns true
```

KEYWORD: "THIS"

When you attach a function as an event handler to an HTML element, then the element becomes the **owner** of the function and so `this` will refer to the element.

```
element.onclick = doSomething;
```

See this link for more info on when `this` refers to an HTML element: http://www.quirksmode.org/js/this.html

KEYWORD: "THIS" IN JQUERY

jQuery uses `this` wrapped in a jQuery selection, so that it can call jQuery methods directly on the selected element.

If you wrap `this` in jQuery - `$(this)` - in your handler, it will refer to the element you have selected using jQuery.

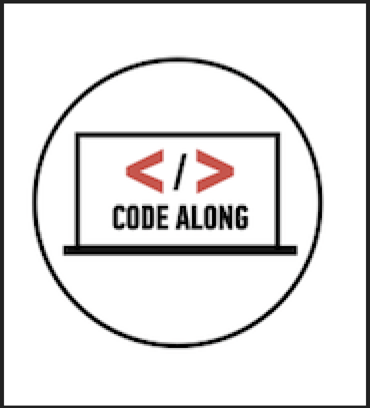
See this codepen: http://codepen.io/wilkom/pen/xZjeWz

KEYWORD: "THIS" IN JQUERY

This is useful when you are applying the same event handler to a group elements and you want each element to work independently.

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

Rule of thumb (ROT): If I don't know what thing will be acted on, then I should consider using "this"



REFACTOR USING THIS

Open *week6_working_files/color_scheme*

DEBUGGING

Why isn't this working?

DEBUGGING

Always start be defining the problem.

- "The image is not moving"
- "None of my code works"

DEBUGGING

This will tell you where to start your hunt

- Image not moving
  - find the code that makes the image move
- None of my code works
  - Syntax error, check console

DEBUGGING: LEVEL 1

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

```
PC: CTRL+SHIFT+J
Mac: COMMAND+OPTION+J
```

Click the error

The location may not be correct but is a good place to start
Ex: Unbalanced brackets or parentheses

DEBUGGING: LEVEL 2

So no red errors but not getting the right answer? Try console.log

Ex:

```
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!

1. Try "Your preferred search engine" search
2. Be ready to clearly articulate the problem (Write out what your problem is)
3. If nothing, ask instructor

</>
CODE ALONG

DEBUG

Open *week6_working_files/debug*