

# ARRAYS AND LOOPS

in JavaScript





# JQUERY PLUGIN REVIEW

Implement a jQuery plugin:

Lesson 11 >

assignment\_solutions >

06\_everything\_together

Check Out

# ASSIGNMENT #1

While the Lecture is Going On

<http://codepen.io/staypuftman/pen/oYmEWJ>

# ARRAYS

```
var fruits = ["banana", "orange", "apple"];
```

- Ordered collection of data types combined into one variable.
- Each item in an array gets assigned an index value.
- Why would we want to do this?

# ARRAYS

```
var fruits = ["banana", "orange", "apple"];
```

```
fruits[0]; // will output "banana"
```

```
fruits[1]; // will output "orange"
```

```
fruits[2]; // will output "apple"
```

-You call array variables using square brackets and putting their index value into the brackets. Notice anything weird?

# [0]

The index value arrays always start with 0, no exceptions. It is never 1.

# CODEALONG

Let's make some arrays and access their  
values

# .LENGTH

```
var fruits = ["banana", "orange", "apple"];  
  
// Notice the length method doesn't use ()  
fruits.length; // will output "3"
```

- Use length to figure out how many items are in your array



# indexOf()

```
var fruits = ["banana", "orange", "apple"];  
  
fruits.indexOf("orange");  
// will output "1"
```

-Use `indexOf()` to see what index value any item in the array has

# .POP() = REMOVE LAST

```
var fruits = ["banana", "orange", "apple"];  
  
fruits.pop();  
// fruits = ["banana", "orange"];
```

-Pop method pops off the LAST item in the array. Can you hear the sound effect when you pop an item off? [BINK]

# \_.SHIFT() = REMOVE FIRST

```
var fruits = ["banana", "orange"];
```

```
fruits.shift();  
// fruits = ["orange"];
```

-Shift method pops off the FIRST item in the array. This one just quietly shifts off into the night.



# **.PUSH() = ADD TO END**

```
var fruits = ["orange"];
```

```
fruits.push("kiwi");
```

```
// fruits = ["orange", "kiwi"];
```

-Push method takes whatever item you have in quotes within the method parenthesis and adds it to the END of the array, creating a new item.

# **.UNSHIFT() = ADD TO BEGINNING**

```
fruits = ["orange", "kiwi"];
```

```
fruits.unshift("cherry");  
// fruits = ["cherry", "orange", "kiwi"];
```

-Unshift method takes whatever item you have in quotes within the method parenthesis and adds it to the BEGINNING of the array, creating a new item.

# `.splice()` REMOVE

```
fruits = ["cherry", "orange", "kiwi"];
```

```
fruits.splice(1,1);
```

```
// 1st number = index value for splice
```

```
// 2nd number = number of items to remove
```

```
// fruits = ["cherry", "kiwi"];
```

-Splice method removes a specific item from the array. First number indicates the index where removal starts, second number indicates total number of items to remove.



# splice() ADD

```
fruits = ["cherry", "kiwi"];
```

```
fruits.splice(1, 0, "pear");
```

```
// 1st value = index value for splice
```

```
// 2nd value = number of items to remove
```

```
// 3rd value = item to be added to array
```

```
// fruits = ["cherry", "pear", "kiwi"];
```

If you add a third value to the splice method, it will add that value into your array.

# REVERSE() ORDER

```
fruits = ["cherry", "pear", "kiwi"];  
  
fruits.reverse();  
// fruits = ["kiwi", "pear", "cherry"];
```

Reverse just switches around the ordering of all items in the array. It doesn't add or delete anything.

# JOIN() ADD

```
fruits = ["kiwi", "pear", "cherry"];  
  
fruits.join(" and ");  
// fruits = "kiwi and pear and cherry";
```

Join will take the value from within the parentheses and combine it with all values in your array to make one big string.



# MULTI-DIMENSIONAL ARRAYS

```
var produce = ["kiwi", "pear", "cherry",  
               "broccoli", "celery", "carrots"];
```

```
produce[1] = ["broccoli", "celery",  
            "carrots"];
```

```
produce[0][2] = "cherry";
```

You can put arrays inside of arrays. Access them with a second set of square brackets (first bracket is the array, second bracket is item).

# CODE ALONG ASSIGNMENT #2

Let's put this together

# PROGRAM FOR CA#2

- 1- Make an array variable of pets
- 2- Output those pets into a select list
- 3- Make a button that displays a picture of each pet



# LOOPS



Check Out

# ASSIGNMENT #3

While the Lecture is Going On

<http://codepen.io/staypuftman/pen/oYmEWJ>

# LOOP

Repetitious program that runs over a defined range.

# WHY LOOPS?

- Loops take advantage of what computers do best: evaluate instructions across organized sets of data very quickly.
- Computers think best in isolated patterns, which is exactly how a loop works.
- Efficient in terms of memory and processor usage



# FOR LOOP

Works similar to an if statement but with more conditions. Three declarations:

- 1- Define variable
- 2- Establish condition for loop to run
- 3- Increment variable

```
for (var i = 0; i < 10; i++) {  
    console.log(i);  
    // outputs 0,1,2,3,4,5,6,7,8,9  
}
```

# FOR LOOP + ARRAYS

Blending the two concepts, we can create powerful programs that move through large amounts of data very quickly.

# BIG IDEA

The major use case for arrays is to create itemized groups of data computers can manipulate (often with loops).

# FOR LOOP + ARRAYS

Same number of declarations but notice the condition in the parenthesis. The array's length limits the amount of loops.

```
var myArray = ["John", "Benjamin", "Victor",  
"Serrao"];  
  
for (var i = 0; i < myArray.length; i++) {  
    console.log(myArray[i]);  
    // outputs "John,Benjamin,Victor,Serrao"  
}
```



# WHILE LOOP

Will evaluate while a condition is met. Very similar to for loops but without incrementing directly in the parenthesis. Notice syntax changes.

```
var myArray = ["John", "Benjamin", "Victor",  
"Serrao"];  
var i = 0;  
  
while (i <= myArray.length) {  
    i++;  
}
```

# FOR VS WHILE LOOP

If in doubt, use a for loop. It's usually the right answer to what you are doing.

5% of the time when you **don't know how many times you want the loop to run**, while loops are best.

# CODE ALONG

## ASSIGNMENT #2

Let's loop through the pets this time

# TRY IT OUT

- Create an array of 5 values and store it as a variable
- Loop through all the values in the array using a for or while loop
  - Inside the loop, write a conditional statement that will print out “I love JavaScript” at the 3rd index value.
  - Inside the loop, write a conditional statement that will print out “I hate JavaScript” at the 4th index value.
  - Inside the loop, write a conditional statement that will print out “The choice is up to you” at the 5th index value.

# NEXT TIME

HTML Forms

this and Function returns

Responsive Design Start

No HW this week

Enjoy Christmas + Hanukkah