# FEWD - Arrays and Scope

**James Gallichio**

## Agenda

- Debugging Techniques
- Arrays
- Scope
- The "this" Keyword

## Debugging

Why isn't this working?

## Debugging

Always start be defining the problem.

- â€The image is not movingâ€
- â€None of my code worksâ€

## 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=ââ;
[âBobâ,âJoeâ].forEach(function(element){
    stringOfNames-=element+â,â;
    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 â€Your preferred search engineâ€ search
2. Be ready to clearly articulate the problem (Write out what your problem is)
3. If nothing, ask instructor

## Arrays

## Problem:

```javascript
var city1 = "New York";
var city2 = "San Francisco";
var city3 = "Sydney";
var city4 = "London";
var city5 = "Tokyo";
// etc...
```

## Arrays

- A special type of data that allows us to store multiple values in a single variable.

## Arrays

```
var cities = ["New York", "San Francisco", "Sydney", "London", "Tokyo"];
```

## Arrays

- Access values by referencing the index number (starts from 0):

```
var cities = ["New York", "San Francisco", "Sydney", "London", "Tokyo"];

var selected = cities[0];          // assigns "New York"
var someOtherCity = cities[3];     // assigns "London"

cities[0] = "Frankston";
cities[2] = "Perth";
```

## Arrays

- Can contain any data type (ie int, float, string, boolean...)
- Can contain other arrays
- Best to keep all elements in an array as the same data type

## Cool, so what can we do with arrays?

## Iterating

- We can iterate through an array and execute code on each item using the "`.each`" method

```
$(cities).each(function(index, element) {
    $("#mainContent").append(element + "<br>");
});
```

## Arrays & Iteration

## Scope

Write this code in your console:

```
var name = "Bob";

function moo() {
    console.log("First the name is: " + name);
    var name = "Jane";
    console.log("Now the name is: " + name);
}

function cow(name) {
    console.log(name);
}

console.log(name);
cow("Stan");
moo();
```

Finished? Try removing `var` from the moo() function

## Scope

**Scope** is the area in which a variable is active.

- **Global** scoped variables are accessible from anywhere in the code
- **Local** scoped variables are only accessible within the function they are declared

# Scope

```
var name = "Bob";         // Global variable

function moo() {
    console.log(name);    // Variable hasn't been defined yet! But the browser knows it's comin
    var name = "Jane";    // Local variable
    console.log(name);
}

function cow(name) {
    console.log(name);    // Local variable
}

console.log(name)
moo();
cow("Stan");
```

## Scope

- The lifetime of a variable starts when it is declared
- Local variables are deleted when the function has finished running
- Global variables are deleted when the page is closed
- Best to avoid global variables if possible

**this**

## this

jQuery: `this` is a special variable that refers to the selected object

## this

How about this code?

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

## this

When should we use `this`?

- When we want to select only the element where the event was fired
- When we want to select the current element in a loop

## Iterating with `this`

```
var cities = ["New York", "San Francisco", "Sydney", "London", "Tokyo"];
```

### Without `this`

```
$(cities).each(function(index, element) {
    $("#mainContent").append(element + "<br>");
});
```

### With `this`

```
$(cities).each(function(index) {
    $("#mainContent").append(this + "<br>");
});
```

## `This` and Scope

- The value of `this` changes depending on where it's used

## `This` and Scope

When used in a method, it refers to the object whose context the function is running in:

```
$(".button").click(function(){
    $(this).toggleClass("selected");
    // "this" refers to the ".button" that was clicked

    $(this).parents(".content").slideToggle();
});
```

## `This` and Scope

- When used in a loop, it refers to the current item in the iteration:

```
var friends = ["Joe", "Tasja", "Allison", "Daniel", "Julia", "Lilly", "Siobhan"];

$(friends).each(function(){
    sendAnnoyingChainEmail(this);
});
```

## Color Scheme Switcher

## Assignment

## Further Reading

- Objects
- Switch statements
- debugger