Javascript

Down the Rabbit Hole

Today’s Agenda

- Javascript for loops with our checkbox list
- Javascript arrays
- Some more tooling around with Javascript
- Javascript Objects
- Javascript Functions – the weird & wild
- The Big Three (window, DOM, data)

for loops!

- Let’s finish looping through our checkbox list
- Remember:

```
for (var i=0; i < something; i++) {
        // Loops!
}
```

Arrays

- an object type that indexes data via index numbers
- *very* similar to Python’s lists
- you can store any other types of objects inside of an array
- you access elements using [] notation

```
var materials = [‘hay’, ‘wood’, ‘hamster’]
materials[0] // will access hay
materials.pop() // Removes hamster
materials.push(‘kittens’) // Makes the list [‘hay’, ‘wood’, ‘kittens’]
```

Arrays cont.

```
// Our list is currently [‘hay’, ‘wood’, ‘kittens’]
materials.length // returns the length of the array
materials.indexOf(‘wood’) // Returns 1
materials.reverse() // Makes list [‘kittens’, ‘wood’, ‘hay’]
materials.shift() // Removes ‘kittens’ from list and returns it
```

Find more cool list things on W3C’s site. http://www.w3schools.com/jsref/jsref_obj_array.asp

Queue vs Stack

materials = [‘hay’,’wood’,’kittens’]

- Stacks are LIFO (Last in First Out)
  - materials.push(‘puppies’)

    [‘hay’, ‘wood’,’kittens’,’puppies’]
  - materials.pop() // Returns ‘puppies’
- Queues are FIFO (First in First Out)
  - materials.push(‘puppies’)

    [‘hay’, ‘wood’,’kittens’,’puppies’]
  - materials.shift() // Returns ‘hay’

List of Materials

Let’s modify our pricing pages so that the list is generated by a list, and when we add items to our page, it adds them to our list.

Objects

Objects group together a set of variables and functions in create a model.
At the end of the day, everything’s an object in Javascript.

- in Python, we defined the model of our objects using Classes
- in Javascript, we can just make Objects.

Making an Object

```
var product = {
    checked: false,
    name: ‘Wood’,
    price: 15,
    stock: 10,

    adjustStock: function(num) {
        return stock -= num;
    }
};
```

Function Expression

```
var adjustStock = function(num) {
    return stock -= num;
}
```

- functions placed where an expression would normally be are called function expressions
- generally, you do not include a name for the function, making it an **anonymous function**
- function expressions are not evaluated until called – meaning you cannot call it before the interpreter reads it

Immediately Invoked Function Expressions (IIFE)

```
inStock = (function() {
    return stock > 0;
}());
```

- “iffy”s are function that are not given a name, as they are called immediately upon interpretation
- the variable they’re assigned to will be set to whatever they return
- Super important! Note the ()’s inside the closing)! This is what makes it an “iffy”!

When to use what

- Iffy’s and Anonymous functions are best for when you need to run code only under very specific circumstances, rather than repeatedly from other parts of the script
- An object’s methods are generally defined as Anonymous functions, as you only call them from within the object
- both can help reduce scope overlap
- Iffy’s are helpful for when you want to wrap code even more securely

this

- a keyword commonly used inside functions and objects, which always refers to one object
- depending on where it’s called depends on its value – it always refers to the default object at your current scope you’re at

- *Global function:* **this** refers to the window object
- *Method inside an Object*: **this** refers to the containing object
- *Function Expression as a method:* **this** refers to the default object of the scope which it’s called

The Big Three Objects

There are 3 sets of built-in objects in Javascript.

- Browser Object Model
  - a model of the browser/window the web page is loaded in
- Document Object Model
  - the DOM is a model of the current web page; it is a child of the Browser Object Model, as well
- Global Javascript Objects
  - these are not a single model; there are the various object types we’ve already been using (String, Number, Boolean, Object,