IO Lab: Object-Oriented Javascript

November 6, 2013

INFO 290TA (Information Organization Lab)
Kate Rushton & Raymon Sutedjo-The
Object-Oriented Programming

Encapsulation: provide access only to an object’s essential details and hide non-essential ones.

Inheritance: inherit the attributes and behavior of another class (to increase reusability).

Polymorphism: make use of an entity in several different forms without affecting the original identity of the entity.

Abstraction: decide on which details are important (and not important) for an object.

http://voices.yahoo.com/features-object-oriented-programming-1607731.html
OOP & Javascript

Object creation patterns (encapsulation)

Code reuse patterns (inheritance)

Javascript Objects

Literal notation

```javascript
var iolabObject = {...}
```

Constructor function

```javascript
function iolabObject() {...}
```

**Properties:** variables attached to an object.

**Methods:** functions attached to an object.
Literal Notation

```javascript
var iolabObject = {
    name: "iolab",
    shout: function() {
        alert(this.name);
    }
};
```

**Property**

// iolabObject.name will return "iolab"

**Method**

// iolabObject.shout() will bring up a dialog box that says "iolab"
Literal Notation

An object literal creates an object that can be immediately used without first having to use the `new` keyword.

However, an object literal cannot implement the basic OOP principles of encapsulation and inheritance.

E.g. say you defined a “gallery” object. If you want to have multiple galleries on the page, you have to duplicate the object multiple times and give it a different name each time.
Constructor Function

```javascript
function iolabObject() {
    this.name = "iolab";
    this.shout = shoutFunction;
    function shoutFunction() {
        alert(this.name);
    }
}
```
Constructor Function

```javascript
var courseOne = new iolabObject();
var courseTwo = new iolabObject();

courseTwo.name = "iolabF13";
```

**Property**

// courseOne.name will return "iolab"
// courseTwo.name will return "iolabF13"

**Method**

// courseOne.shout() will bring up a dialog box that says "iolab"
// courseTwo.shout() will bring up a dialog box that says "iolabF13"
Constructor Function

A constructor creates a **blueprint** of objects, not the object itself.

Methods & properties are declared with a **this** prefix.

New objects are initialized with the **new** keyword.

As such, we can create multiple instances of an object easily.

E.g. say you defined a “gallery” object. If you want to have multiple galleries on the page, you can initiate those instances with the **new** keyword:

```javascript
var articleSlideshow = new gallery();
var popularPhotos = new gallery();
```
Final Project
Final Project

Open-ended as long as it’s somehow related to IO/IR.

Start thinking about ideas! Post them on Piazza or pitch it next week (after P 3 presentations).

Remember, this is the final project, so be creative & think bigger.
Next Class
Next Class

P 3 presentations

7 minutes per group

< 1 min. group introduction

5 mins. project summary & demo

2 mins. Q&A

Time limit will be **strictly enforced**

Final project pitches and group formation at the end of class