# Lecture 6

## What we discussed in the last class

- **How does ‘this’ work?**

‘this’ gives features of dynamic scoping in javascript. In a nutshell this references the object itself, so naturally it needs to bind to someone to give refer to itself.

Try opening your node terminals and just write this. It prints a long object, now define a variable x by var x = 5; Now write this.x, it would print 5. What is this pointing to in this case? The global object, all variables defined in the global context are essentially properties of the global object.

```
function foo() {
	this.val  = 5;
	this.func = function () {
		console.log(this.val);
	}
}

var o = new foo();
// execute o.func to print o.val
o.func();

// store this function into a new variable x
var x = o.func;
// execute x, now clearly x store the function func
x(); // prints undefined
```

Why does execution of x print undefined?

Because x is exactly the function func. But x is not binded to any object explicitly hence it is bound to the global object instead and clearly the global object has no val, so when the function stored in x tries to do console.log(this.val) it prints undefined.

**NOTE :**
*So for you to reference something using ‘this’ in a function, the function call should be **bound to the object you want to reference using ‘this’**, otherwise this would fallback to the global object if it exists.*

*Binding a function to an object can be accomplished using bind, call, apply or simply if the function is a property of that object as in the above case, doing **o.func() binds func to o.***

- **Bind, Call, Apply**

You are expected to read about these functions on MDN instead to become more self reliant, but still I would give a brief introduction to get you started.

Extending the example given above, I have three ways to have the x() call work as intended.

```
var y = x.bind(o);
x.call(o);
x.apply(o);
```

Bind returns a function that is bound to the provided object.
Call and Apply execute the functions bound to the provided object.

Bind, call and apply also take additional parameters. Bind, call and apply take extra parameters that are passed to functions as arguments.

## Quiz in the next class on Basics of Javascript.

## Next class snapshot

- Understanding the event loop.

## Important resources

- **JS Expert :** http://eloquentjavascript.net/

Keep on reading eloquent javascript (JS Expert Link) whenever you have time, with the level of understanding people have currently, I have serious doubts if people are really reading.