Client Scripting
Advanced Functions
What is in scope at each point?

```javascript
var aGlobal;

function f1(aParameter) {
    var aLocal = 0;
    function f11(bParameter) {
        var bLocal = 0;
        for (var bCtr = 1; bCtr < 10; bCtr++) {
            bLocal += bCtr;
        }
    }
    for (var aCtr = 1; aCtr < 10; aCtr++) {
        aLocal += aCtr;
    }
    f11(aLocal);
}
```

This code is at: http://www.andrew.cmu.edu/course/67-328/examples/scope.html
This code is at: http://www.andrew.cmu.edu/course/67-328/examples/scope.html
var aGlobal;

function f1(aParameter) {
  var aLocal = 0;
  function f11(bParameter) {
    var bLocal = 0;
    for (var bCtr = 1; bCtr < 10; bCtr++) {
      bLocal += bCtr;
    }
  }
  for (var aCtr = 1; aCtr < 10; aCtr++) {
    aLocal += aCtr;
  }
  f11(aLocal);
}

This code is at: http://www.andrew.cmu.edu/course/67-328/examples/scope.html
var aGlobal;

function f1(aParameter) {
    var aLocal = 0;
    function f11(bParameter) {
        var bLocal = 0;
        for (var bCtr = 1; bCtr < 10; bCtr++) {
            bLocal += bCtr;
        }
    }
    for (var aCtr = 1; aCtr < 10; aCtr++) {
        aLocal += aCtr;
    }
    f11(aLocal);
}

What is in scope at each point?

This code is at: http://www.andrew.cmu.edu/course/67-328/examples/scope.html
Review Scope

What is in scope at each point?

var aGlobal;

function f1(aParameter) {
    var aLocal = 0;

    function f11(bParameter) {
        var bLocal = 0;
        for (var bCtr = 1; bCtr < 10; bCtr++) {
            bLocal += bCtr;
        }
    }

    for (var aCtr = 1; aCtr < 10; aCtr++) {
        aLocal += aCtr;
    }

    f11(aLocal);
}

This code is at: http://www.andrew.cmu.edu/course/67-328/examples/scope.html
Review Scope

What is in scope at each point?

```javascript
var aGlobal;

function f1(aParameter) {
    var aLocal = 0;
    function f11(bParameter) {
        var bLocal = 0;
        for (var bCtr = 1; bCtr < 10; bCtr++) {
            bLocal += bCtr;
        }
    }
    for (var aCtr = 1; aCtr < 10; aCtr++) {
        aLocal += aCtr;
    }
    f11(aLocal);
}
```

This code is at: http://www.andrew.cmu.edu/course/67-328/examples/scope.html
Review Scope

What is in scope at each point?

```javascript
var aGlobal;

function f1(aParameter) {
  var aLocal = 0;
  function f11(bParameter) {
    var bLocal = 0;
    for (var bCtr = 1; bCtr < 10; bCtr++) {
      bLocal += bCtr;
    }
  }
  for (var aCtr = 1; aCtr < 10; aCtr++) {
    aLocal += aCtr;
  }
  f11(aLocal);
}

This code is at: http://www.andrew.cmu.edu/course/67-328/examples/scope.html
```
If you are fuzzy with this, review:

- Section 3.10 of JavaScript: The Definitive Guide
JavaScript Classes

• Work as a table
• For each person:
  – Explain your class
  – Explain your method extension of the String class
  – Pick one String method in your group that was most interesting and/or clever
Method Chaining

• If a method does not return anything, then have it return this.
  – What is this?
• In this way, the returned value can be used to chain additional methods.
function Calculator (initValue) {
    this.value = initValue;
}
Calculator.prototype.add = function(addend) {
    this.value += addend;
    return this;
}
Calculator.prototype.sub = function(subtrahend) {
    this.value -= subtrahend;
    return this;
}
Calculator.prototype.equals = function() {
    return this.value.toString();
}

• You can then chain:
  var c = new Calculator(0);
  c.add(5).add(17).sub(8).equals();
• Method chaining is used a lot in jQuery.

This code is at: http://www.andrew.cmu.edu/course/67-328/examples/calculator.html
Method Chaining Exercise

• Working in pairs
• Create a class with methods that can chain.
  – One constructor
    • The constructor should also be chainable
  – Three separate methods that chain
• OK to use one of your solutions to today's homework
• Demonstrate it with an example.
Asynchronous Programming

• Real-life metaphors for **asynchronously**
  – Events
    • My daughter's goofy dog: Barks wildly when she hears a doorbell
      – Even if doorbell is on the TV
      – Even if she is outside the door and her walker rang the bell
    • Otherwise she can deal with other tasks, like sleeping or eating.
    • She implements a listener function: on hearing doorbell (action), bark wildly
  – Timer
    • Set an alarm clock to wake up at 8:00am
    • A timer is a type of event, but programmatically generated.
    • You implement a listener function: upon alarm, wake up
  – Asynchronous processes
    • In order to have a pizza delivered:
      – call and order the pizza
      – give instructions for how to ring your apartment once they arrive at the building
    • These instructions are like a callback function, a function you give the process (pizza person) to do when the task is completed
    • Between ordering pizza and answering door, you can do other things
Asynchronous Programming

- Client-side (browser) **asynchronously**
  - User Events
    - E.g. onclick, onmouseover, onfocus
    - And lots more
  - Timer
    - setTimeout - do callback after some given time
    - setInterval – do callback after every given time interval
  - Asynchronous processes
    - E.g. make a request to the server (AJAX) and run a callback function when the reply is completed
```javascript
function tick() {
    console.log("tick ");
    setTimeout(tock, 1000);
}

function tock() {
    console.log("tock ");
    setTimeout(tick, 1000);
}
```

Callback function is passed to `setTimeout`

"When you hit 1000 ms, call this function."

This code is at: http://www.andrew.cmu.edu/course/67-328/examples/tick.html
Anonymous function as a callback

- Anonymous functions are often used as a callback
- I.e. functions defined in-line, without names
- This version works identically to the last one

```javascript
function tick() {
    console.log("tick ");
    setTimeout(function(){
        console.log("tock ");
        setTimeout(tick, 1000);
    }, 1000);
}
```

This code is at: http://www.andrew.cmu.edu/course/67-328/examples/tick2.html
Async / Closure exercise

• **Without using a global variable,** have tick and tock produce the following:
  
tick 0
  tock 1
  tick 2
  tock 3
  tick 4
  tock 5
  ...

© Joe Mertz - Distributed Application Development
Would the following work?

function tick() {
    var tcount = 0;
    console.log("tick "+ tcount++);
    setTimeout(function(){
        console.log("tock "+ tcount++);
        setTimeout(tick, 1000);
        }, 1000);
    }

This code is at: http://www.andrew.cmu.edu/course/67-328/examples/tick3a.html
Closure exercise

• Thinking it through...
  – You need a variable that is in scope for tick and tock
  – But it cannot be in the global scope
  – There are 2 possible scope ranges:
    • global
    • function
  – Therefore you need a function variable that is in scope to both tick and tock
  – This implies that tick and tock must be nested within a 3rd function (e.g. startClock).
  – The tick and tock counter (e.g. tcount) can be a function variable of this outer function, and by closure it will be within scope of tick and tock (even after the outer function has completed executing).

• Continue working on the exercise...