Practical Functional JavaScript

Oliver Steele
Ajax Experience
Wednesday, 1 October 2008

Disclaimer

This isn't the actual slide deck. It's a placeholder that I've wedged into Keynote format so I can make a PDF out of it, and doesn't (yet) include code examples.

Check back after the talk for the actual deck – I'll do the work of incorporating the examples then.

Teasers

- AJAX is all about waiting for someone*, and remembering what you were going to do when they got back to you.
- Functions : interactions :: objects : domains
- You didn't really want threads anyway. (Most of the time.)

* user, web server, other server, wall clock, plugin

About Me

		graphics	languages	writing	using
Entrepreneurial & Consulting	BrowseGoods Style&Share Fansnap Webtop Calendar	✔	✔		✔
Laszlo Systems	OpenLaszlo	✔	✔	✔	✔
Apple Advanced Technology Group	Dylan (programming language)		✔	✔	
Apple System Software	Skia (graphics library)	✔		✔	

About *You*

Raise your hand if you know*:

Closures
Ruby / Smalltalk
XHR / AJAX
Frameworks (Prototype / jQuery / …)
Threads

* none of these are required; this just helps me calibrate the talk

Agenda

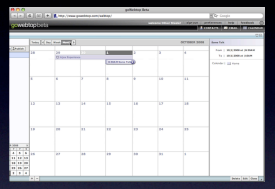
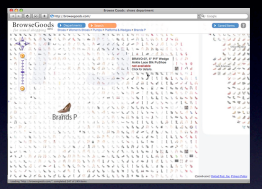
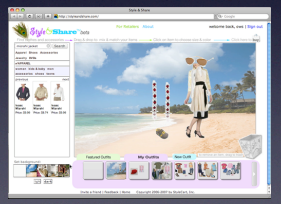
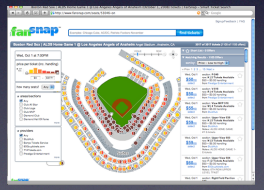
- The Problem
  - MVC on the Client
  - Forgetting what you were going to say
- Fundamentals
  - Closures (review)
  - *Making* Functions
  - *Decorating* Functions
- Callbacks
  - Threaded Callbacks
  - Registered Callbacks
  - Order & Serializing
  - Retries, Guards, Timeouts
- Background Processing
  - Use Cases
  - How To
  - Pros and cons vs. Gears
- Q&A

Non-Agenda

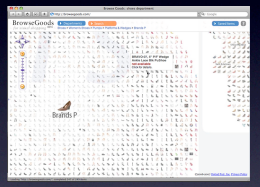
- Comet, Bayeux, Gears
- Frameworks*
- Theory (this isn't your Monad or CPS fix)
- Security (standard practices apply)

* This talk should help you understand their *implementation* and *use*, but doesn't explore their *APIs* in any depth

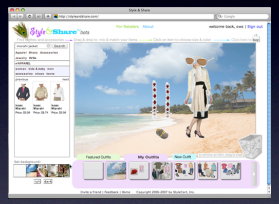
Some Examples

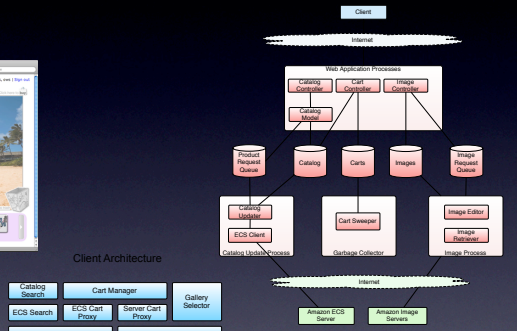





BrowseGoods

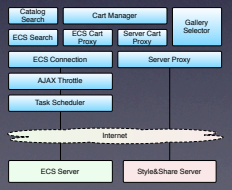


Style & Share

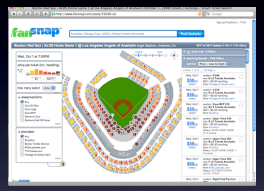




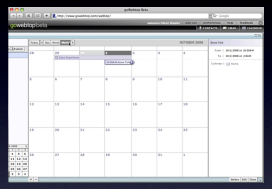
Client Architecture

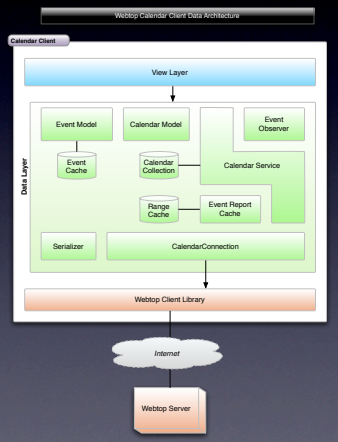


Fansnap

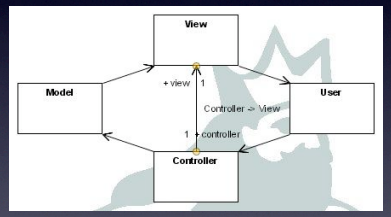


Webtop Calendar

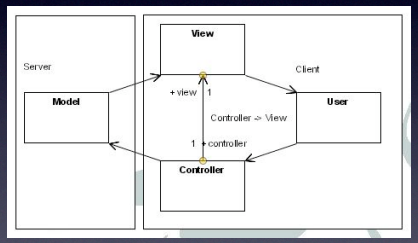




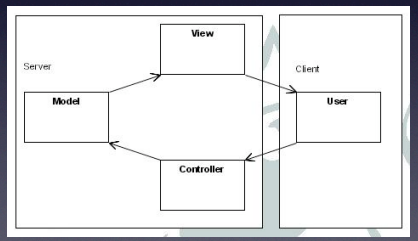
The Problem: Web MVC



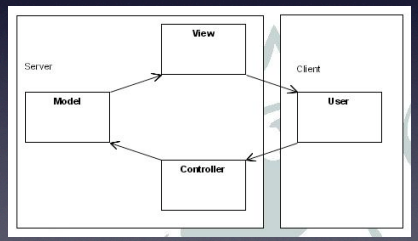
Waiting on the Server



Waiting on the User



Waiting on the Client



Function Fundamentals

What is a Function?

- Math: rule that maps inputs to outputs
- Computer science: abstracted computation with effects and outputs
- Software engineering: one of several units for documentation, testing, assertions, and analysis
- Source code: unit of source text with inputs and outputs
- Runtime: invocable parameterized behavior

Callbacks

```
function doit() {
    $.post('/request', {}, callback);
}

function callback(x) {
    alert('received ' + x);
}
```

Run

Callbacks

```
function doit2() {
    $.post('/request', {}, callback);
    function callback2(x) {
        alert('received ' + x);
    }
}
```

Run

Callbacks

```
function doit3() {
    $.post('/request', {}, function callback2(x) {
        alert('received ' + x);
    });
}
```

Run

Callbacks

```
function doit4() {
    $.post('/request', {}, function (x) {
        alert('received ' + x);
    });
}
```

Run

Making Functions

```
function makeOne() {
    return function() { return 1; }
}

function makeN(n) {
    return function(n) { return n; }
}

function makeAddOne() {
    return function(x) { return x + 1; }
}
```

Decorating Functions

```
function twice(fn) {
    return function(x) {
        return fn(fn(x));
    }
}

var addTwo = twice(makeAddOne);
console.info(addTwo(10));
```

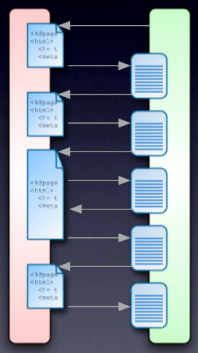
Registering Functions

```
var FnTable = {};
function register(name, fn) { FnTable[name
function tableMethod(name) { return FnTabl

register('+1', makeAddOne);
register('+2', twice(makeAddOne));

console.info(tableMethod('+1')(10));
console.info(tableMethod('+2')(10));
```

Guards

- (example)

Callbacks



Server-side web application

Callback Challenges

- Chained Callbacks
- Queues and Priority
- Throttling
- Caching
- Timeouts
- Retry and Failover
- Conjunctive-Trigger Callbacks
- Conditional Callbacks
- Outdated Responses

Queues and Priority

- Case: Prioritize outgoing requests
- Case: Multiple queues
- Case: Jumping the queue
- (Samples)

Throttling

- Case: Throttle outgoing requests
- Case: Server load and adjustable throttles
- (examples)

Timeouts

- Case: Suspending

Caching

- Case: Caching responses (easy)
- Case: Merging multiple requests for the same resource (hard)
- Case: Invalidating the cache based on subsequent requests
- (examples)

Timeout and Retry

- Case: Timeout independently of XHR
- Case: Retry
- Case: Failover
- (examples)

Conjunctive Triggers

- Case: Waiting for one of several responses (easy)
- Case: Waiting for all of several responses (hard)
- (examples)

Stale Responses

- Case: Suspending response handlers

Background Processing

Divided Work

- Rolling computation by hand
- (examples)

Sequences

- Rolling computation with a helper
- (examples)

Comparison with threads

- Interstate traffic and interprocess state
- Yielding from inside out

Q&A