The Promised Land:
Untangling Async Spaghetti Code

Presented by Trevor Burnham at FluentConf 2012
Preface

The Compleat History of JavaScript
(in about a minute)
1995: JavaScript discovered
2001: pg says fuhgeddaboutit

“I would not even use Javascript, if I were you... Most of the JavaScript I see on the Web isn’t necessary, and much of it breaks.”

(“The Other Road Ahead,” 2001)
2004: Practical use discovered
2006: Solid libraries discovered

...And probably others!
2008: Good parts discovered
2009: Additional uses discovered
2011: JS taught as first language

http://www.codecademy.com/
2012: Knowing JavaScript has become a superpower!
So, everything is perfect now.

Thanks for coming!

Follow me @trevorburnham and @coffeescript
mainWindow.menu("File", function(err, file) {
    if(err) throw err;
    file.openMenu(function(err, menu) {
        if(err) throw err;
        menu.item("Open", function(err, item) {
            if(err) throw err;
            item.click(function(err) {
                if(err) throw err;
                window.createDialog('DOOM!', function(err, dialog) {
                    if(err) throw err;
                    ...
                });
            });
        });
    });
});

“I love async, but I can’t code like this.”
Andy McDermott

The Pyramid of Doom

A LOST KING'S PAST HOLDS A DEADLY SECRET.

A NOVEL
Why is this unique to JavaScript?

Thread.sleep(0)
Some people, when confronted with a problem, think, "I know, I'll use threads," and then two they hav erpoblesms.
Good design patterns FTW!

https://github.com/documentcloud/backbone/wiki/Projects-and-Companies-using-Backbone
jQuery Promises
What is a Promise? (Abstractly)

A **Promise** is an object representing a task with two potential outcomes: success (“resolved”) or failure (“rejected”).
History of Promises

• Added in jQuery 1.5...
• ...which borrowed the concept from Dojo (Deferred)...
• ...which borrowed it from Python’s Twisted...
• ...which borrowed it from the E language (1997)...
• ...which borrowed it from... y’know what? Just read http://en.wikipedia.org/wiki/Promise_(programming)
What is a Promise? (Concretely)

Instead of taking a callback, an async function can return a **Promise**, an object that success and failure callbacks can be attached to.

```javascript
var promise = $.get('/mydata');
promise.done(onSuccess);
promise.fail(onFailure);
promise.state();  // 'pending'
```
Making Promises

In jQuery, a Promise is a “read-only” copy of a Deferred:

```javascript
var deferred = $.Deferred();
asyncRead(function(err, data) {
    if (err) {
        deferred.reject();
    } else {
        deferred.resolve(data);
    }
});
var Promise = deferred.promise();
```
Why are Promises Awesome?

Reason #3: Promises help you organize code more clearly.

```javascript
plugin.fetch = function() {
  var fetchPromise = $.get(plugin.url);
  promise.fail(plugin.failHandler);
  promise.done(plugin.successHandler);
  return fetchPromise;
};

var fetchPromise = plugin.fetch();
fetchPromise.done(...);
```
Why are Promises Awesome?

**Reason #2:** Promises make it trivial to combine async tasks.

```javascript
function afterAjaxAndFade() { ... }
var ajaxPromise = $.get('/mydata');
var fadePromise = $button.fadeOut().promise();
$.when(ajaxPromise, animationPromise)
  .done(afterAjaxAndFade);
```
Why are Promises Awesome?

**Reason #1:** ...

```javascript
var passwordPromise = $.get('/password');
passwordPromise.done(function(data) {
    var loginPromise = $.post('/login', data);
});

// I wish I could bind to loginPromise here!
```
Why are Promises Awesome?

**Reason #1:** Promises let you see the future!

```javascript
var passwordPromise = $.get('/password');
var loginPromise = passwordPromise.pipe(
    function(data) {
        return $.post('/login', data);
    }
);
loginPromise.done(...);
```
Remember this?

```javascript
mainWindow.menu("File", function(err, file) {
  if(err) throw err;
  file.openMenu(function(err, menu) {
    if(err) throw err;
    menu.item("Open", function(err, item) {
      if(err) throw err;
      item.click(function(err) {
        if(err) throw err;
        window.createDialog('DOOM!', function(err, dialog) {
          if(err) throw err;
          ...
        });
      });
    });
  });
});
```
var menuFilePromise = mainWindow.menu("File");
var openFilePromise = menuFilePromise.pipe(function(file) {
    return file.openMenu();
});
var menuOpenPromise = openFilePromise.pipe(function(menu) {
    return menu.item("Open");
});
var itemClickPromise = menuOpenPromise.pipe(function(item) {
    return item.click()
});
var createDialogPromise = itemClickPromise.pipe(function() {
    return window.createDialog("Promises rock!");
});
createDialogPromise.fail(function(err) { throw err; });
PIPED PROMISES

DEFEATED PYRAMID OF DOOM!
What if you’re not using jQuery?

A self-contained implementation of jQuery Promises is available:

https://github.com/Mumakil/Standalone-Deferred

Or if you prefer the CommonJS Promises/A spec, use q:

https://github.com/kriskowal/q
Async JavaScript is the first full book I've seen dedicated to a key topic in JavaScript development today: how to deal with concurrency and concurrent tasks without going crazy!" —Peter Cooper
Questions?

Follow me @trevorburnham and @coffeescript