Promises, Promises: Unlocking the Power of jQuery's Deferreds

Brian Klaas
Johns Hopkins Bloomberg School of Public Health
bklaas@jhsph.edu
@brian_klaas

Code for today's session:

github.com/brianklaas/
jquerySD2014-promises

The Problem

Is it done?

Try `setTimeout()`

```
function processForm(data) {
  makeAjaxCall(data);
  setTimeout(showSuccessMsg, 3000);
}

function showSuccessMsg() {
  ...
}
```

Resolve parallel processes with counters

```
var stepsNeededToBeDone = 0;

function makeDinner(data) {
  $.ajax( "/makePizza" ).done(function() {
    stepsNeededToBeDone++; }).fail(function() { alert("error"); });
  $.ajax( "/makeSalad" ).done(function() {
    stepsNeededToBeDone++; }).fail(function() { alert("error"); });
  $.ajax( "/openBeer" ).done(function() {
    stepsNeededToBeDone++; }).fail(function() { alert("cry!"); });

    setTimeout(checkForCompletion, 2000);
}

function checkForCompletion() {
  if (stepsNeededToBeDone == 3) {
    dinnerIsReady();
  } else {
    setTimeout(checkForCompletion, 2000);
  }
}
```

Get into callback hell

```
function makeDinner(data) {
  $.ajax( "/makePizza" )
  .done(function() {
    $.ajax( "/makeSalad" )
    .done(function() {
      $.ajax( "/openBeer" )
      .done(function() { dinnerIsReady(); })
      .fail(function() { alert("Noooooooo"); });
   }).fail(function() { alert("No spring mix"); });
  }).fail(function() { alert("Pizza burned"); });
}
```

Wouldn’t this be a whole lot nicer?

```
when(
  $.ajax("/makePizza"),
  $.ajax("/makeSalad"),
  $.ajax("/openBeer"),
).then(dinnerIsReady);
```

The solution for asynchronous work:

Promises

What is a promise?

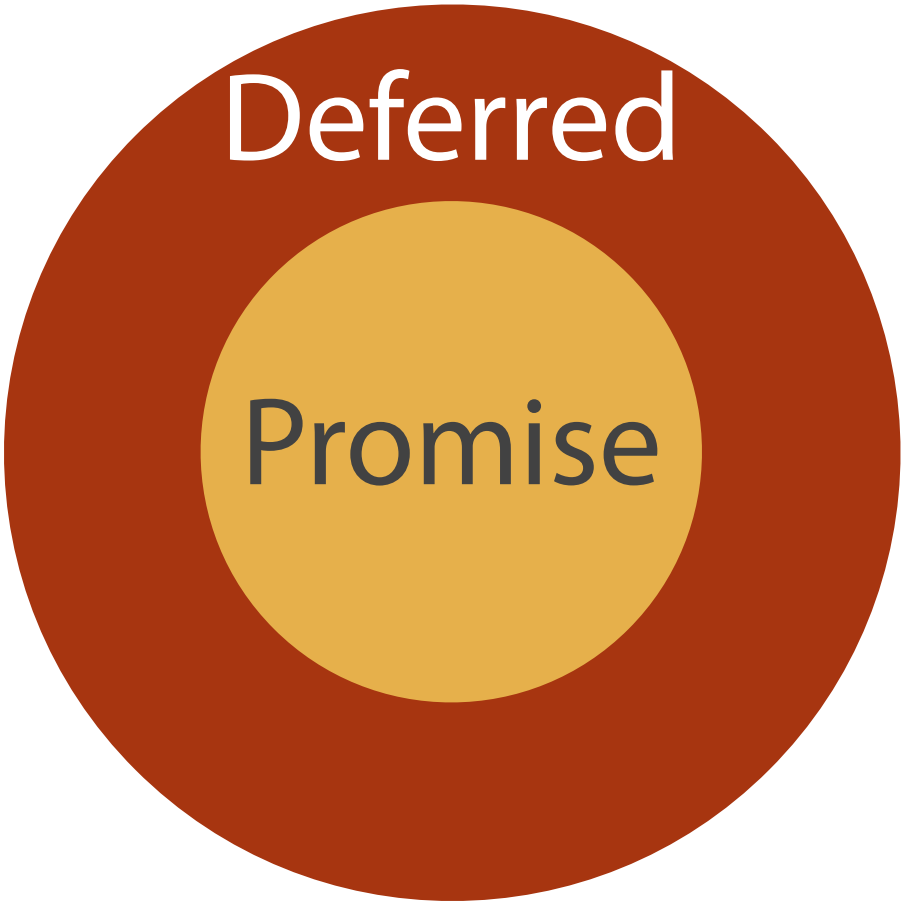
Or, more to the point: what is a deferred?

```
$(document).ready()
```

Promises enable developers to think and express asynchronous code in a more synchronous way.

A promise is an object that represents a one-time event, typically the outcome of an async task like an AJAX call.

You can make multiple async calls and defer them to a single promise.


Deferred
Promise

promise = result that is not yet known

deferred = work that is not yet finished

A deferred has methods
that allow its owner
to resolve or reject it.

At first, a promise is in a pending state.

Eventually, it's either resolved (done) or rejected (failed) by the deferred.

You can attach callbacks to the promise, which will fire when the promise is resolved or rejected.

Promises make it easy to say: "When all of these things have happened, do this other thing."

```
when(
  $.ajax("/makePizza"),
  $.ajax("/makeSalad"),
  $.ajax("/openBeer"),
).then(dinnerIsReady);
```

http://promises-aplus.github.io/promises-spec/

jQuery Promise Methods

An Example Deferred

```
console.log("Using a base Deferred");
var deferred = new $.Deferred();

console.log(deferred.state());  // "pending"
deferred.resolve();
console.log(deferred.state());  // "resolved"
deferred.reject(); // no effect, the Promise was already resolved
console.log(deferred.state());  // "resolved"
```

An Example Promise

```
console.log("Now using Promises");
var deferred = new $.Deferred();
var promise = deferred.promise();

console.log(promise.state());  // "pending"
deferred.reject();
console.log(promise.state());  // "rejected"
```

An Example Promise: wait()

```
function wait(ms) {
  var deferred = $.Deferred();
  setTimeout(deferred.resolve, ms);
  return deferred.promise();
}

wait(1500).then(function () {
  console.log("We waited 1500ms");
});
```

`$.done()` = promise is resolved

`$.fail()` = promise is rejected

But we're not limited
to just `$.done()` and `$.fail()`

$.when()

For waiting on multiple deferreds to resolve

$.when() returns a new promise that obeys these rules:

- When *all* of the given promises are resolved, the new promise is resolved.
- If *any* of the given promises is rejected, the new promise is rejected.

$.when() Example

```
$.when(
  promiseOne,
  promiseTwo
)
  .done(function () {
    console.log('promiseOne and promiseTwo are done');
})
  .fail(function () {
    console.log('One of our promises failed');
});
```

$.then()

The core of the Promises/A+ spec

$.then() Example

```
promiseOne.then(function () {
   console.log('promiseOne done');
   promiseTwo.then(function () {
      console.log('promiseTwo done');
      console.log('All done');
   });
});
```

A Cleaner $.then()

```
promiseOne.then(function () {
  console.log('promiseOne done');
}).then(function () {
  console.log('calling promiseTwo');
  return promiseTwo;
}).then(function () {
  console.log('All done');
});
```

You need to return a promise from then()
if you want to chain it.

Even better $.then():
Chain using named functions

Animating with Promises

Syncing animations is not fun.

jQuery's `animate()` toolbox uses promises

Parallel Animation

```
$.when(
  fadeOut('#divToFadeOut'),
  fadeIn('#divToFadeIn')
).done(function () {
  console.log('Parallel animation finished');
  $('p').css('color', 'red');
});
```

Chained Animation

```
fadeOut('#divToFadeOut')
  .then(function (el) {
    console.log("Fading in");
    fadeIn(el);
  }).then(function (el) {
    console.log("Chained animation finished");
  });
```

Solution:
Chain using named functions.

Promises and AJAX

In which you discover that you are already using promises.

jQuery returns promises from all of its AJAX methods.

```
var jqxhr = $.ajax( "/example" )
    .done(function() { alert("success"); })
    .fail(function() { alert("error"); })
    .always(function() { alert("complete"); });
```

jqXHR.success(), jqXHR.error()
are deprecated!

Use jqXHR.done(), jqXHR.fail(), and
jqXHR.always() instead.

http://api.jquery.com/jQuery.ajax/

```
var jqxhr = $.ajax( "/example" )
    .then(function() { alert("Finished!"); });
```

```
promise.then(doneCallback, failCallback, alwaysCallback)
```

```
jqXHR.then(
  function(data, textStatus, jqXHR) { ... success code... },
  function(jqXHR, textStatus, errorThrown) { ...failure code... }
);
```

```
$.when($.ajax("/page1"), $.ajax("/page2"))
  .then(successFunc, failureFunc);
```

Displaying content only after it loads

```
$.when(
  getLatestNews(),
  getLatestTweets(),
  showAsyncLoadedContent()
).then(function(){
  console.log( "UI fully loaded." );
}).fail(function(){
  console.log( "Something went wrong loading content!" );
});
```

Promises and UI/App Notifications

`.progress()`

Allows you to attach callbacks that are executed when `notify()` is called on the deferred.

Progress Meter Example

```
var fileProgress = $.Deferred();

// Called by the notify() method of a deferred object
fileProgress.progress(function(valToShow) {
  $("#progBar").val(valToShow);
});
```

Beyond jQuery

Backbone

```
this.model.save().then(null, function(obj)
{
  console.log(obj.responseText);
});
```

```
$.when(collectionA.fetch(),collectionB.fetch())
.done(function(){
    //success code here.
});
```

AngularJS: $q

Promise–based APIs

```
Parse.User.logIn("user", "pass").then(function(user) {
  return query.find();
}).then(function(results) {
  return results[0].save({ key: value });
}).then(function(result) {
  // the object was saved.
});
```


node
JS
™

ahead
aligator
api–chain
assure
augur
avow
bond
branches
cancellation
clues
concurrent
covenant
deferred–queue
faithful
faithful–exec
futures
holdup
jasync
kew

Q

Q

Can exchange promises with jQuery.

Available as an AMD and Common.js module, and in Bower.

Implements `then()`, `fail()`, `fin()` and more.

Processing Credit Card Payments with Q

```
var deferred = Q.defer();
httpRequest.post({
    url: "https://mypaymentprocessor.com",
    auth: { user, pass },
    json: params
}, function(error,response,body){
    if(body.status_code>=400){
        deferred.reject(body.reason);
        return;
    }
    // Successful Requests
    deferred.resolve(body);
});
return deferred.promise;
```

The Future of Futures


Go Do

Thank you!

Brian Klaas
Johns Hopkins Bloomberg School of Public Health
bklaas@jhsph.edu
@brian_klaas
www.iterateme.com
github.com/brianklaas

Resources Used in Building this Presentation

- http://net.tutsplus.com/tutorials/javascript-ajax/wrangle-async-tasks-with-jquery-promises/
- http://blog.mediumequalsmessage.com/promise-deferred-objects-in-javascript-pt1-theory-and-semantics
- http://blog.mediumequalsmessage.com/promise-deferred-objects-in-javascript-pt2-practical-use
- http://addyosmani.com/blog/jquery-1-7s-callbacks-feature-demystified/
- http://msdn.microsoft.com/en-us/magazine/gg723713.aspx
- https://gist.github.com/ThomasBurleson/1910025
- http://eng.wealthfront.com/2012/12/jquerydeferred-is-most-important-client.html

Resources Used in Building this Presentation

- http://blog.parse.com/2013/01/29/whats-so-great-about-javascript-promises/
- http://www.html5rocks.com/en/tutorials/async/deferred/
- http://coenraets.org/blog/2013/04/building-pluggable-and-mock-data-adapters-for-web-and-phonegap-applications/
- http://stackoverflow.com/questions/13148356/how-to-properly-unit-test-jquerys-ajax-promises-using-jasmine-and-or-sinon
- http://www.jonnyreeves.co.uk/2012/unit-testing-async-javascript-with-promises-and-stubs/
- http://tech.pro/blog/1402/five-patterns-to-help-you-tame-asynchronous-javascript
- https://hacks.mozilla.org/2013/07/so-you-wanna-build-a-crowdfunding-site/