promises
the way of the angular

AngularJS Portugal
meetup.com/AngularJS-Portugal/
twitter.com/angularjspt
Google+ community

/$ whoami

Vitor Fernandes
vmlf01@gmail.com

- 15+ years developer
- CTO with Gotv Media Software since 2002
- Creator of AngularJS Portugal group

Visual Basic 5, Microsoft WebTv ASP/Jscript, .Net Framework 1 through 4.5 desktop apps, Titanium Mobile Javascript Mobile apps, AngularJS last 2 months

what are promises

What are promises?

A promise is a container that holds or will hold a value

WHAT?!?

It's a trust thing!

With promises, you are always guaranteed to get one and only one value back (even if it's a failure)

Trust me :)

what are promises

What do they do?

Promises are a way of thinking synchronously and doing asynchronously.

What is it good for?!

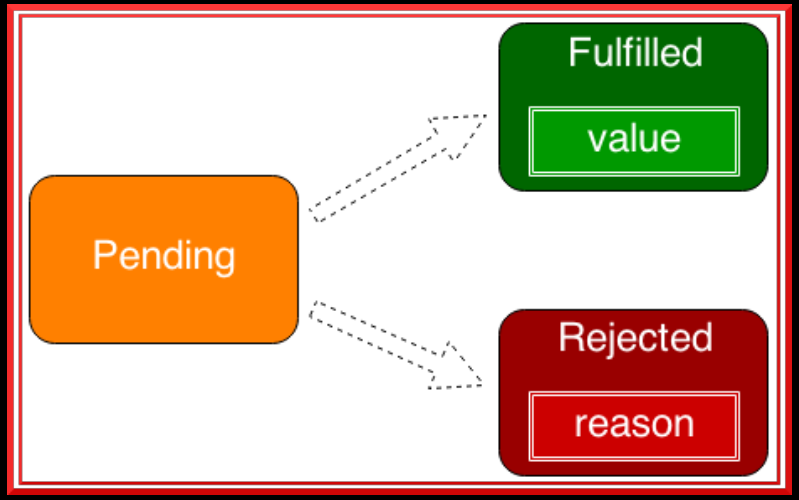
Unlike war, absolutely everything.

Say it again!

promise states

A promise can be:

- Pending
- Fulfilled
- Rejected



Once a Promise is fulfilled or rejected, it’ll remain in that state forever.

promise guarantees

promiseForResult.**then**(onFulfilled, onRejected);

- Only one of **onFulfilled** or **onRejected** will be called.
- **onFulfilled** will be called with a single *fulfillment value* (⇔ return value).
- **onRejected** will be called with a single *rejection reason* (⇔ thrown exception).
- If the promise is already settled, the handlers will still be called once you attach them.
- The handlers will always be called asynchronously.

the promised land in angular, domenic denicola

programming 101: algorithms

Dinner out:

1. How many people?
2. Which restaurant?
3. Book restaurant
4. Show up on time
5. Eat

programming 101: algorithms

Dinner out:

Well that's all fine and dandy,
but what about the other stuff?

In reality, you cannot block waiting for each step to complete. You need to go about your life!

(Web) programming is async.

parallel execution

Dinner out:

1. How many people?
2. Which restaurant?

Done at same time

3. Book restaurant
4. Show up on time
5. Eat

handle exceptions

Dinner out:

1. How many people?
2. Which restaurant?
3. Book restaurant

What if not available?

4. Show up on time
5. Eat

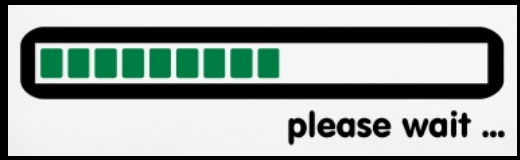
don't block, give feedback

Dinner out:

1. How many people?
2. Which restaurant?
3. Book restaurant
4. Show up on time

Table not ready, wait a bit

5. Eat



finally

Dinner out:

1. How many people?
2. Which restaurant?
3. Book restaurant
4. Show up on time

5. Eat

promises in AngularJS

- Implemented in AngularJS $q service
- Inspired by Kris Kowal's Q library
- Contains only the most important features
- Integrated with AngularJS's digest/event loop

$q

```
function aSimplePromise() {
    var deferedTask = $q.defer();

    deferedTask.resolve(
        'I always keep my promises!'
    );

    return deferedTask.promise;
}
```

$q

The deferred API

```
var deferred = $q.defer()
deferred.resolve(value)
deferred.reject(reason)
deferred.notify(status)
var promise = deferred.promise
```

$q

The promise API

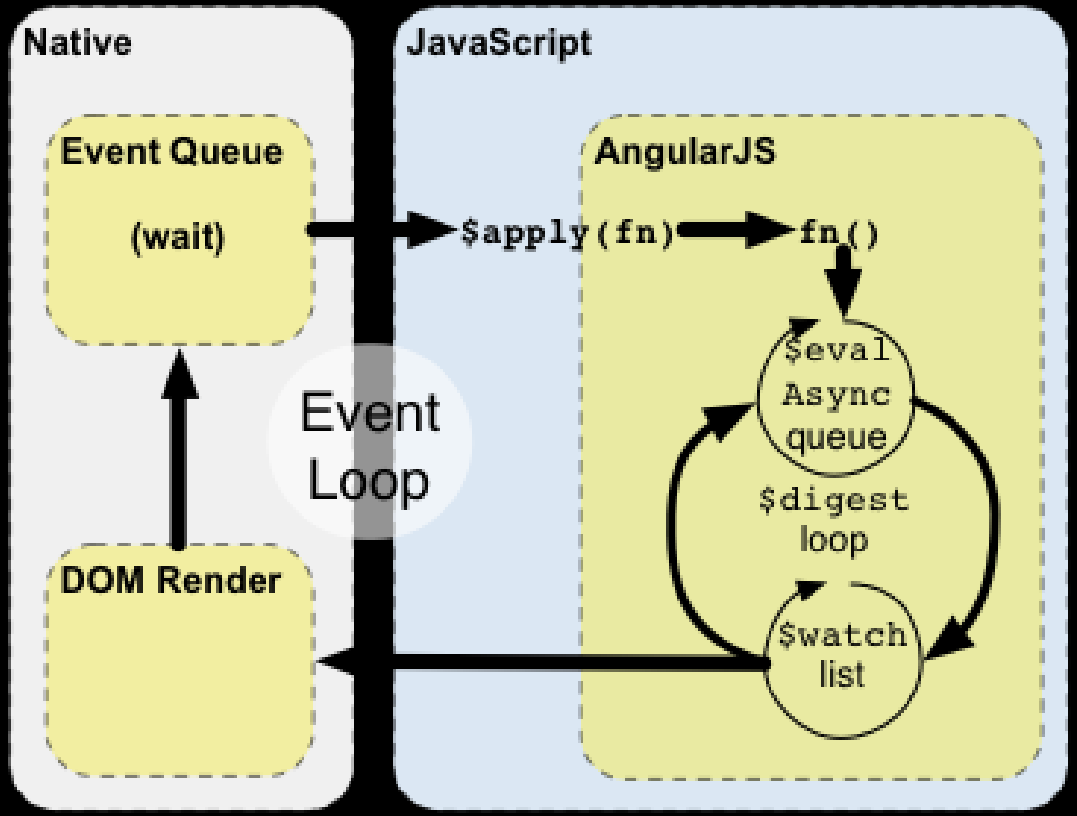
promise.**then**(onFulfilled, onRejected, onNotify)
promise.**catch**(onRejected)
promise.**finally**(callback)

$q

The Helpers

```
var promise = $q.when(value)
var promise = $q.all([arrayOfPromises])
```

AngularJS digest cycle



the promised land in angular, domenic denicola

$q resolution inside digest cycle


```
function MyController($scope, $http) {
    $scope.text = "loading";

    $scope.doThing = function () {
        $http.get("somefile.json").then(function (response) {
            $scope.text = response.data;
        });
    };
}
```

// Works! Angular's promises are integrated into the digest cycle

the promised land in angular, domenic denicola

The Tao of Angular

Promises are all around in AngularJS!

$http
$resource
$route
$timeout
...

the 5 commandments

{5..9}

- **then**() returns a promise
- Callback return values fulfill their promises
- Callback errors reject their promises
- Value/reason passthru if callback missing
- Callbacks can return promises

promises. the basics, from promises/A+, luke smith

promises problems

Can we live without them?

...

Why would you?

- Wasted promises → Garbage collection
- Silent fails

why promises are awesome

- Cleaner method signatures
- Uniform return/error semantics
- Easy composition
- Easy sequential/parallel join
- Always async
- Exception-style error bubbling

callbacks, promises, and coroutines (oh my!), domenic denicola

how do promises work?

Wait, wait, wait...

But how does it work?

how do promises work

Well, eh...

Js promises are just

callbacks

in hidding!

Did you think it was magic???

function Promise(fn)

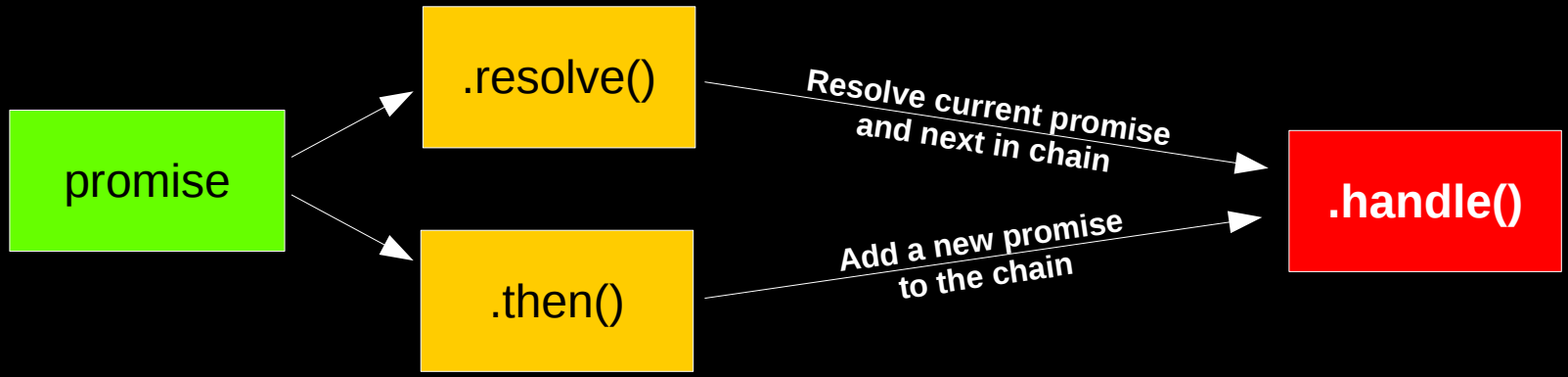
```
var state = 'pending';
var value;
var deferred = null;

function resolve(newValue) {
    value = newValue;
    state = 'resolved';

    if(deferred) {
        handle(deferred);
    }
}
```

```
function handle(handler) {
    if(state === 'pending') {
        deferred = handler;
        return;
    }
    if(!handler.onResolved) {
        handler.resolve(value);
        return;
    }
    var ret = handler.onResolved(value);
    handler.resolve(ret);
}
```

function Promise(fn)

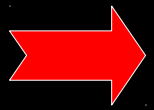
```
this.then = function(onResolved) {
    return new Promise(function(resolve) {
        handle({
            onResolved: onResolved,
            resolve: resolve
        });
    });
};

fn(resolve);
```

promise call flow

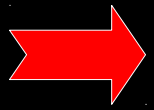


function handle(handler)

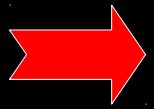
```
function handle(handler) {
    if(state === 'pending') {
        deferred = handler;
        return;
    }
    if(!handler.onResolved) {
        handler.resolve(value);
        return;
    }
    var ret = handler.onResolved(value);
    handler.resolve(ret);
}
```



If promise state is pending, store handler to call it back later when promise resolves



If there is no onResolved callback for this promise, then take its value and passthru to next promise in chain



Call this promise onResolved handler

Then, resolve next promise in chain with the return value

references

Promises/A+ Specification
http://promises-aplus.github.io/promises-spec/

Callbacks, Promises, and Coroutines (oh my!), Domenic Denicola
http://www.slideshare.net/domenicdenicola/callbacks-promises-and-coroutines-oh-my-the-evolution-of-asynchronicity-in-javascript

The Promised Land in Angular, Domenic Denicola
http://www.slideshare.net/domenicdenicola/the-promised-land-in-angular

Promises, Luke Smith
http://www.slideshare.net/drprolix/promises-16473115

JavaScript Promises ... In Wicked Detail, Matt Greer
http://www.mattgreer.org/articles/promises-in-wicked-detail/

Thanks

promises

the way of the angular

Vitor Fernandes
vmlf01@gmail.com

AngularJS Portugal
meetup.com/AngularJS-Portugal/
twitter.com/angularjspt
Google+ community