AngularJS Promises

by Jeremy Zerr

Blog: http://www.jeremyzerr.com
LinkedIn: http://www.linkedin.com/in/jrzerr
Twitter: http://www.twitter.com/jrzerr
Github: https://github.com/jrzerr
Plunker: http://plnkr.co/users/jrzerr

Promises

□ What is a promise?

□ What features do AngularJS promises have?

□ What would you use a promise for?

□ Demo using promises for Image Preloading

Code and Demo

- □ Code and demo are all on my Plunker page
  - □ Basic Promises
  - □ Multiple Promises
  - □ Image Preloading Service

AngularJS Promises

Do that first…
then do this

What is a promise?

- A promise is a way to manage asynchronous actions performed within your app.
- When the action is done, you can process the result.
- Like saying: When you are done doing this, then do that

Promise = $.ajax?

- We are familiar with the concept of an AJAX callback from jQuery.
  - $.ajax(url).done(function(response) {});
- $.ajax returns a jQuery promise, which is similar, but different than AngularJS promises.
- The AngularJS promise system extends this pattern.

AngularJS Promise Features

- It is generic, not only limited to API calls.
- You can have any code run and use a promise to control program flow when it completes.
- You can chain actions to take after a promise is completed. “When you are done doing that, then do this, and this, and…”
- You can separately handle success, errors, and notifications.
- You can group several promises together into one promise. “After you are done doing those 8 things, then do this”
- If you have a promise, but it has already been resolved, you don’t have to wait. “Oh, you are done doing those 8 things, then do this now”

Promise System

- The Promise system is run in AngularJS using the $q service. It is part of core AngularJS, just inject it!
- It consists of two parts:
  - Deferred API
  - Promise API

Deferred API

- □ Create a deferred object by $q.defer()
- □ resolve(value) - success
- □ reject(reason) - fail
- □ notify(value) - progress
- □ promise property

Promise API

- The promise object is a property on your deferred object
- then(successCallback, errorCallback, notifyCallback)
- catch(errorCallback) - shorthand
- finally(callback) - for observation

Angular $q

- defer() - get a deferred object
- reject(reason) - like a throw in JS
- when(value) - wrapping another object as a promise
- all(promiseArray) - create a new promise from multiple promises

Simple Promise Demo

- Let’s look at some code for a simple demo of using a promise to preload a single image.
- Most typical code pattern is:
  - Create deferred object.
  - Pass along to your long-running process.
  - Save the promise and add a then() to control what is done on completion.
- Plunker Simple Promise Demo

What Are Some Uses For Promises

You are probably already using them!

Promise Use Cases

- API calls
  - When the API call is done, do something with the data.
  - Do you use $http? $http.get() returns a promise, which we typically decorate with a .then() which is part of the Promise API. $resource also uses it, it uses $http internally.

Promise Use Cases

- Preloading images
    - Loading each image can take different amounts of time to load. Avoid your content shifting, or images flashing, as several images are loaded into your page.
    - Load images into browser Image() objects before tying them to DOM. Makes for a great opportunity for a service and directive pair.

Promise Use Cases

- Workflow
  - You can chain responses to create a flexible workflow. Think of many small functions processing some data instead of one big one.
  - Could pass the promise around and components could independently attach a then() handler. Great way to decouple your design.

Promise Use Cases

- Pre-requisite Data/Services before rendering a view
  - ui-router has a resolve property for a state (in ui-router, a state roughly is the same as a url)
  - If you add promises to the resolve list, the promises must resolve before the view will be rendered
  - This is a great place to add fetching absolutely necessary data for the view, like the User, or Configuration options
  - ui-router can even have resolve promises dependent on others so you can add workflow before your view is rendered

Promise Use Cases

- Notifications of a long-running algorithm
  - You can send notifications of progress to improve user experience.
  - If you have a certain number of steps in an algorithm or process, can keep the user updated of the REAL progress. Not just a spinner or estimated time till download (why is it stuck at 0????)

Preview of Image Preloading

Image Preloading

- Our project:
  - We have several images to load on the screen.
  - There can be long delays to retrieve the images.
  - We do not want the user to see the images pop up as they are loaded, we want all images to show at once, completely loaded.
  - Plunker Promises Demo

Image Preloading - Part 2

- □ Our improvement:
  - □ We would like the preloading to be in a service.
  - □ We would like to pick the number of images we want to see.
  - □ We want to use ui-router and use the resolve feature.
  - □ Plunker Promises + UI-Router Demo

Thanks!

by Jeremy Zerr

Blog: http://www.jeremyzerr.com
LinkedIn: http://www.linkedin.com/in/jrzerr
Twitter: http://www.twitter.com/jrzerr
Github: https://github.com/jrzerr
Plunker: http://plnkr.co/users/jrzerr