ANGULARJS OVERVIEW

INTRODUCTIONS

WHO WE ARE

- James Churchill - Developer/App Dev Practice Manager
- Ken Howard - Front End Developer
- Jason Domogalla - Front End Developer

WHO CSG PRO IS

- Portland based consultancy
- 30 (or so) employees/contractors
- 22 years in business
- Custom Application Development
- Business Intelligence

NG-CONF EXTENDED EVENT

MARCH 5TH AND 6TH

TICKETS: $20 AT EVENTBRITE

AGENDA

- (Optional) Brief History of Web Apps
- Our Evolution
- Why AngularJS???
- App Demos
- "Hello World" Walkthrough
- Q&A
- Resources

BRIEF HISTORY OF WEB APPS

COLLECTION OF PAGES

"THICK SERVER" OR "THIN CLIENT"

- Full page response with each request
- Both presentation and data are sent to the client every time
- Data is packaged with each page response
- Routing and view rendering on the server
- Various tricks to maintain state
  - Session cookie
  - Hidden frame

SINGLE PAGE APPS (SPA)

"THICK CLIENT" OR "THIN SERVER"

- Single initial page load
- All HTML/CSS/JS is loaded on the intial request (typically)
- AJAX is used to load data as needed (driven by user events)
- Routing and view rendering on the client
- State is maintained in the browser (in JS)
- Increased complexity in the client
- Mitigated by using popular patterns like MVC and DI

SPA COMPONENTS

SPA COMPONENT CATEGORIES

- Templating
- Controllers
- Routing
- Local Storage

SPA COMPONENT "LEGO" PIECES

Combine libraries like "lego" pieces to build applications

- Knockout.js - Templating and Controllers
- Crossroads.js - Routing
- RequireJS - Module Loading

SPA FRAMEWORKS

POPULAR SPA FRAMEWORKS

- AngularJS
- Ember.js
- Meteor

SPA FRAMEWORK PROS

- Provide a single, cohesive solution for building SPAs
  - Fewer choices to make
    - Evolving APIs
  - Strong communities

SPA FRAMEWORK CONS

- Steep learning curve
- Frequent API changes
- Sometimes poor documentation
- Monolithic solution
- Lock-in

OUR EVOLUTION

CLASSIC ASP

- Page based
- Light abstractions
- Application, Request, Response, Server, and Session
- VB Script - Interpreted on the server
- Some JS

ASP.NET WEB FORMS

- Page based
- Attempted to make the "stateless" web stateful
- Heavy abstractions
- New versions of ASP abstractions
- Introduction of page and control events
- Control abstractions

ASP.NET WEB FORMS

- C# - Compiled and executed on the server
- More JS, AJAX, and the introduction of jQuery
- Difficult to unit test

ASP.NET MVC

- Response to Ruby on Rails
- Route based
- Embraced the web as being "stateless"
- Fewer abstractions
- Brings the developer closer to HTTP and response/request

ASP.NET MVC

- Introduction of MVC pattern
- C# - Compiled and executed on the server
- Even more JS, AJAX, and jQuery
- Easier to unit test

ANGULARJS AND WEB APIS

- Single page applications
- Stateful, richer UI, "desktop" like applications
- Server simplifies to a web API
- More easily allows division of labor between the front and back end

WHY ANGULARJS???

- Very popular
- Strong community
- Future looks promising
- Backed by Google

ANGULARJS KEY FEATURES

- Declarative two-way data binding
- Ability to extend HTML via Directives
- Fully unit testable

APP DEMOS

"HELLO WORLD" WALKTHROUGH

Heavily inspired by Dan Wahlin's AngularJS in 20ish Minutes

See https://www.youtube.com/watch?v=tnX0-i7944M

QUICK AND DIRTY

- Bootstrapping the app with "ng-app"
- Data Binding with "ng-model"
- Data with "ng-init"
- Repeating data with "ng-repeat"
- Filtering the data with a filter

IMPROVING THE DESIGN

- Modules
- Controllers with "ng-controller"
- Routing with ngRoute module
- Route configuration
- Views with "ng-view"
- Project structure

Q&A

RESOURCES

- AngularJS Site
- AngularJS YouTube Page
- John Papa's AngularJS Style Guide
- Code School AngularJS Free Course