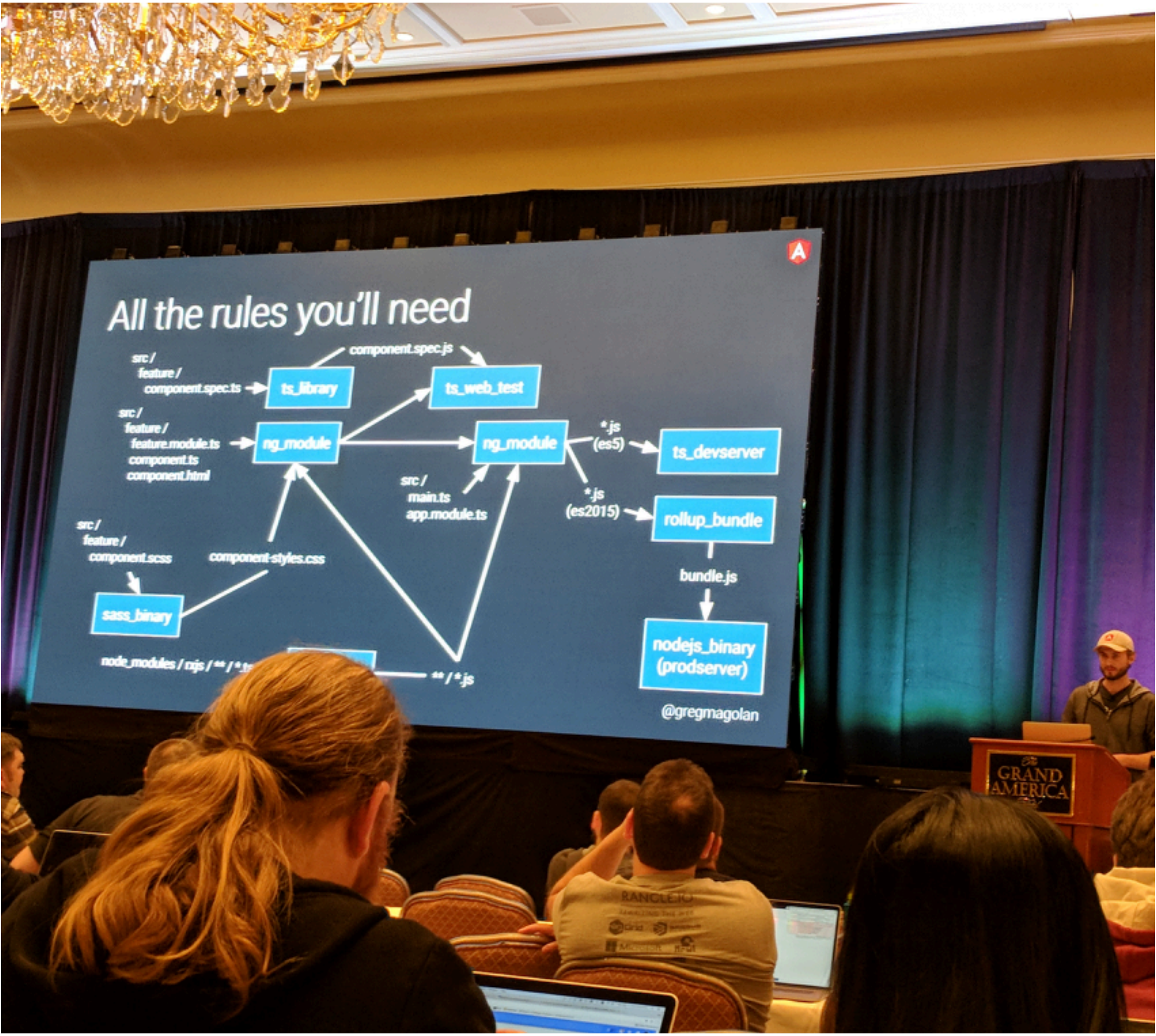
Bazel

I think we would need to approach this as an entire group. It is too much work for just the UI project.

Benefits of Bazel

- Fast at Scale
- Full-Stack - inputs are files, structure is the application, not “how” but “what”
- Builds depend on the size of the change, not the size of the app
- Customizable
    - Structure of the app
    - Paths for output
    - Plugins for the languages and tools
- Bazel rules
    - Input file → action → output fileIncremental Builds
    - Perform build steps/actions
    - Chained together rules and abstract complexities
- Roadmap:
    - Now via AngularLabs This Year: Ergonomics, Performance, Code-splitting
    - Later → converge with Angular CLI
- http://g.co/ng/abc http://g.co/ng/abc http://github.com/alexeagle/angular-bazel-example
- http://github.com/alexeagle/angular-bazel-example/wiki



All the rules you'll need

src /
feature /
component.spec.ts

ts_library

component.spec.js

ts_web_test

src /
feature /
feature.module.ts
component.ts
component.html

ng_module

ng_module

src /
main.ts
app.module.ts

*.js
(es5)

ts_devserver

*.js
(es2015)

rollup_bundle

bundle.js

nodejs_binary
(prodserver)

src /
feature /
component.scss

component-styles.css

sass_binary

node_modules / rxjs / ** / *.ts

** / *.js

@gregmagolan

Angular at Large Organizations

Promote Best Practices Using Tooling

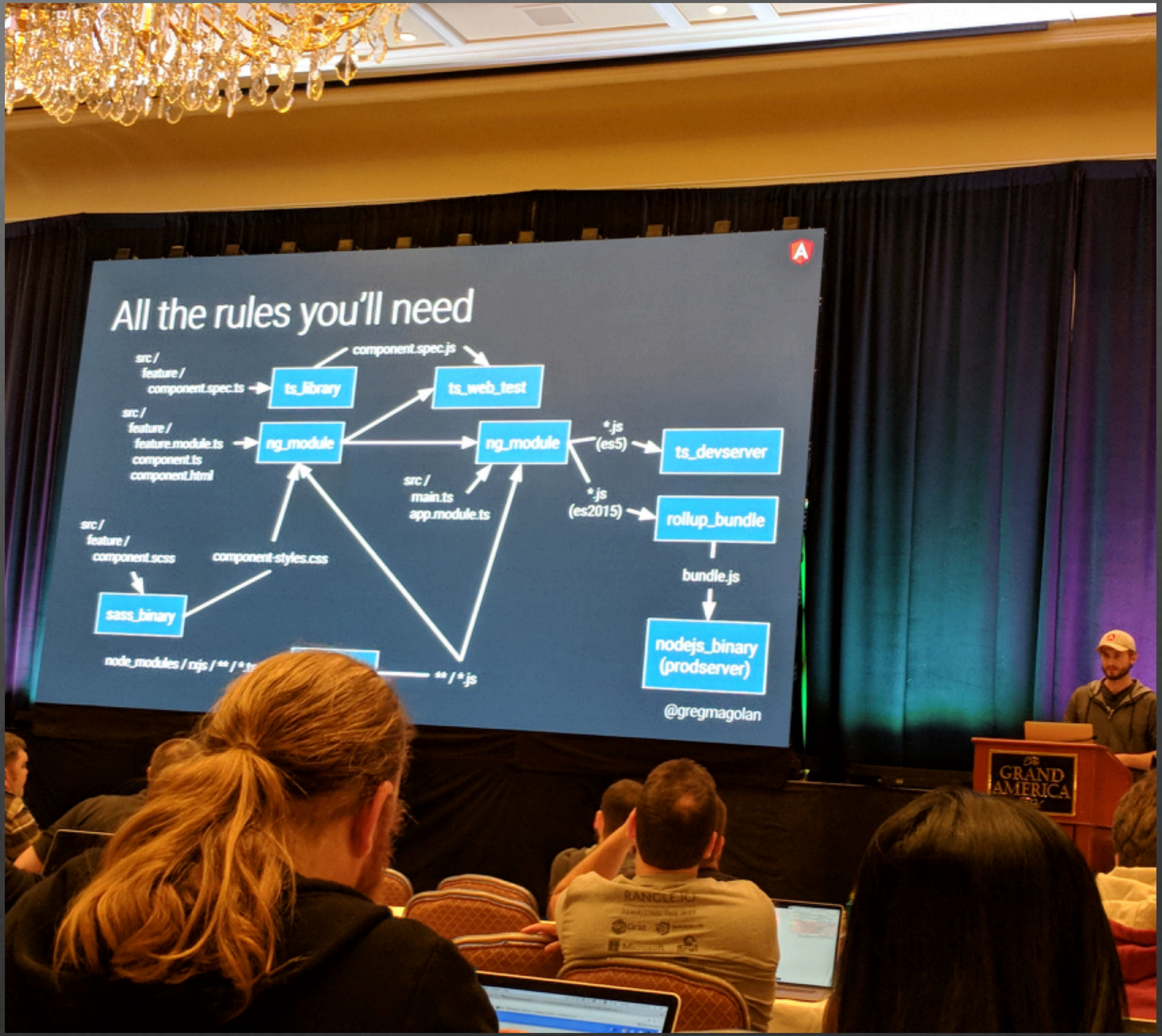
- Only run tests against code that has been changed “–untracked”
- npm yarn dept-graph allows you to visualize dependenciesPromote Best Practices Using Tooling
- Typically: An out-of-sync wiki page no one reads, code reviews that don’t check much
- To Fix
  - schematics and custom schematics (yarn workspace-schematics)
  - data-access-lib, yarn dep-graph,
  - allows users to have a basic boiler plate
  - Automate everything that can be automated (use Prettier)
  - Checklist: https://drive.google.com/drive/folders/13OnVqbFvz-w-dDc2FQ9_Yv2eXmrhBl5X

Bazel

I think we would need to approach this as an entire group. It is too much work for just the UI project.

Benefits of Bazel

- Fast at Scale
- Full-Stack - inputs are files, structure is the application, not “how” but “what”
- Builds depend on the size of the change, not the size of the app
- Customizable
    - Structure of the app
    - Paths for output
    - Plugins for the languages and tools
- Bazel rules
    - Input file → action → output fileIncremental Builds
    - Perform build steps/actions
    - Chained together rules and abstract complexities
- Roadmap:
    - Now via AngularLabs This Year: Ergonomics, Performance, Code-splitting
    - Later → converge with Angular CLI
- http://g.co/ng/abc http://g.co/ng/abc http://github.com/alexeagle/angular-bazel-example
- http://github.com/alexeagle/angular-bazel-example/wiki


All the rules you'll need
src / feature / component.spec.ts
ts_library
component.spec.js
ts_web_test
src / feature / feature.module.ts component.ts component.html
ng_module
ng_module
*.js (es5)
ts_devserver
src / main.ts app.module.ts
*.js (es2015)
rollup_bundle
bundle.js
nodejs_binary (prodserver)
src / feature / component.scss
component-styles.css
sass_binary
node_modules / rxjs / ** / *.ts
** / *.js
@gregmagolan
GRAND AMERICA