Angular Advanced Performance

océ

A CANON COMPANY

Peter Kassenaar –
info@kassenaar.com

“Performance” has many faces

Build / load time

performance

Run time performance

1. Tips on Load time performance - checklist

- *Optimize your builds* by using `ng build`
- Use the `--prod` flag for:
  - AOT-compiling
  - Uglifying
  - Minifying
  - Removal of source maps
  - Bundling (by using WebPack)
  - Tree shaking (enabled by default)
  - And (much!) more
  - https://github.com/angular/angular-cli/wiki/build

- Use *Lazy Loading* in your app
  - Don’t load anything that is not immediately necessary
  - At the very minimum use

    `PreloadingStrategy: PreloadAllModules`
  - Consider writing a custom loading strategy
  - https://angular.io/guide/lazy-loading-ngmodules

- Consider using *Server Sided Rendering* (SSR)
  - Compiled app is served to the browser – fast startup time
  - User interaction is captured and stored/cached until the complete app is loaded.
  - Apps can be indexed, identified and analyzed by Google Bot
  - Can be tricky to set up!
  - Use Angular Universal module for SSR: https://angular.io/guide/universal

- Update Angular CLI and Angular Packages regularly
  - Newer builds typically provide smaller bundles, faster startup times etc.
  - Ivy Renderer will be included by default (near future)

```
npm install -g @angular/cli

ng update
```

On third party libs:

- Use RxJS 6 or higher
- Use a lib that is compatible with tree shaking
- Don’t include everything. Only the stuff you need
  - i.e: create custom builds of Bootstrap, jQuery, Lodash, etc, if you decide to use these
- Use vanilla JavaScript wherever possible.
  - Often you don’t need lodash, jQuery anymore to perform basic tasks
- Use gzip compression on your backend!
  - Compress the files on the server

- Compress your images
  - Consider using a tool like TinyPNG to compress images from your IDE
  - https://marketplace.visualstudio.com/items?itemName=andi1984.tinypng
  - Other image compression tools are available: https://www.google.nl/search?q=image+compressor
- Remove unused fonts from the app

2 - Runtime performance - checklist

- Use `ChangeDetection.OnPush` to avoid unneccessary evaluation of component trees
  - This is the #1 runtime performance tip. Often overlooked!
- Detach the change detector completely if you want full control over CD
  - `this.cdr.detach` in `ngAfterViewInit() { … }`
  - `this.cdr.detectChanges()` when you want to perform CD on demand.

- Use *pipes* to format stuff in the UI.
  - Don’t let CD handle this (as this can become very expensive quite fast!)
  - https://codeburst.io/angular-tips-the-importance-of-pipes-49be3b1e99e7
- *Don’t do computations* in the View/UI
  - DOM is slow
  - Use TypeScript for that

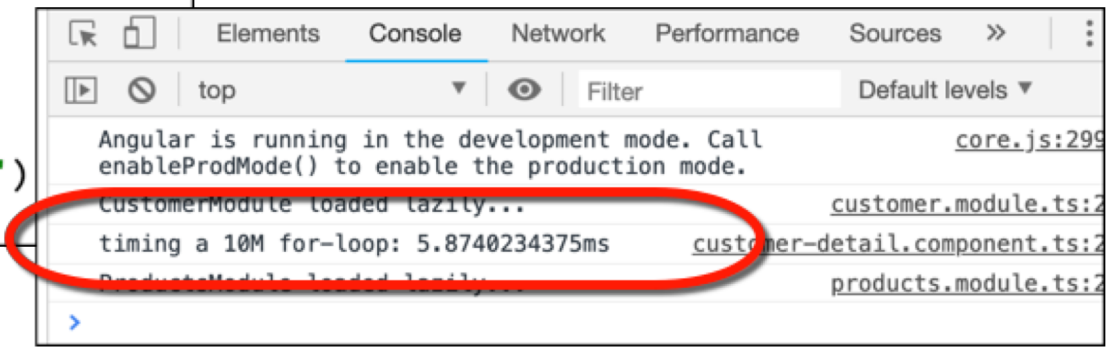
- Remember to *unscubscribe* your observables to avoid memory leaks
  - Or let Angular `async` pipe handle that for you
- If you have multiple subscribers to a source, use the `share()` operator
  - This avoids the processing of duplicate data among subscribers.
  - `this.http.get<any>(‘http://some/endpoint').pipe(share());`

Q: "How to measuring response times for angular actions?"

A: You can use `console.time()` for that

https://alligator.io/js/console-time-timeend/

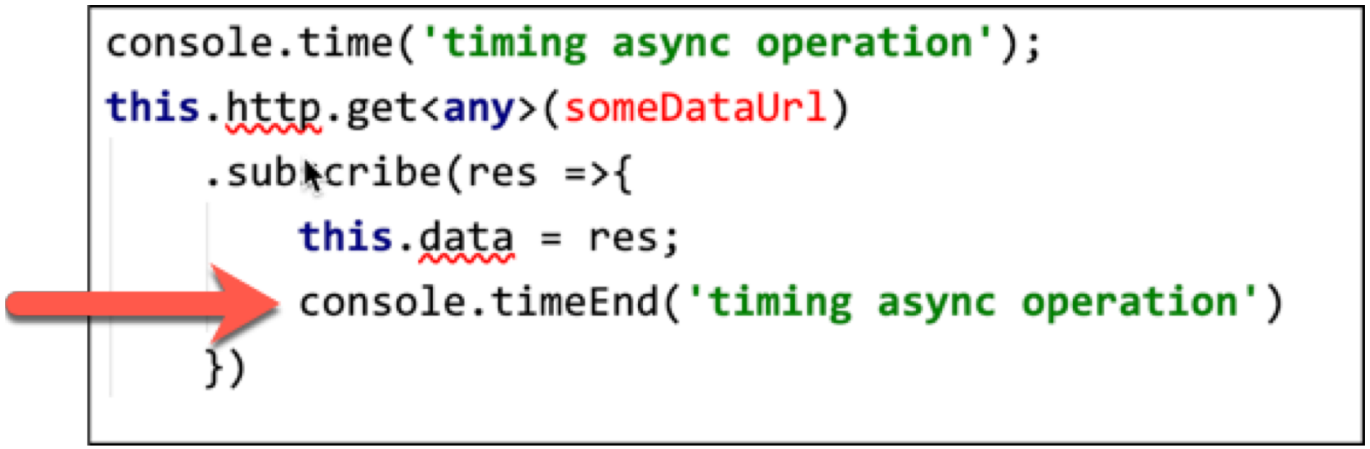
```
// timing the performance of an Angular action
console.time('timing a 10M for-loop');
for (let i = 0; i < 10000000; i++) {
    i++
}
console.timeEnd('timing a 10M for-loop')
```



Timing async operations

Beware - when using an async operation, be sure to place the `console.timeEnd()` *inside* the callback.

Not right after it!



More info

- https://blog.thoughtram.io/angular/2017/02/02/making-your-angular-app-fast.html
- https://www.youtube.com/watch?v=ybNj-id0kjY – Minko Gechev –Optimizing an Angular application
- https://github.com/mgechev/angular-performance-checklis
- https://medium.com/@spp020/44-quick-tips-to-fine-tune-angular-performance-9f5768f5d945