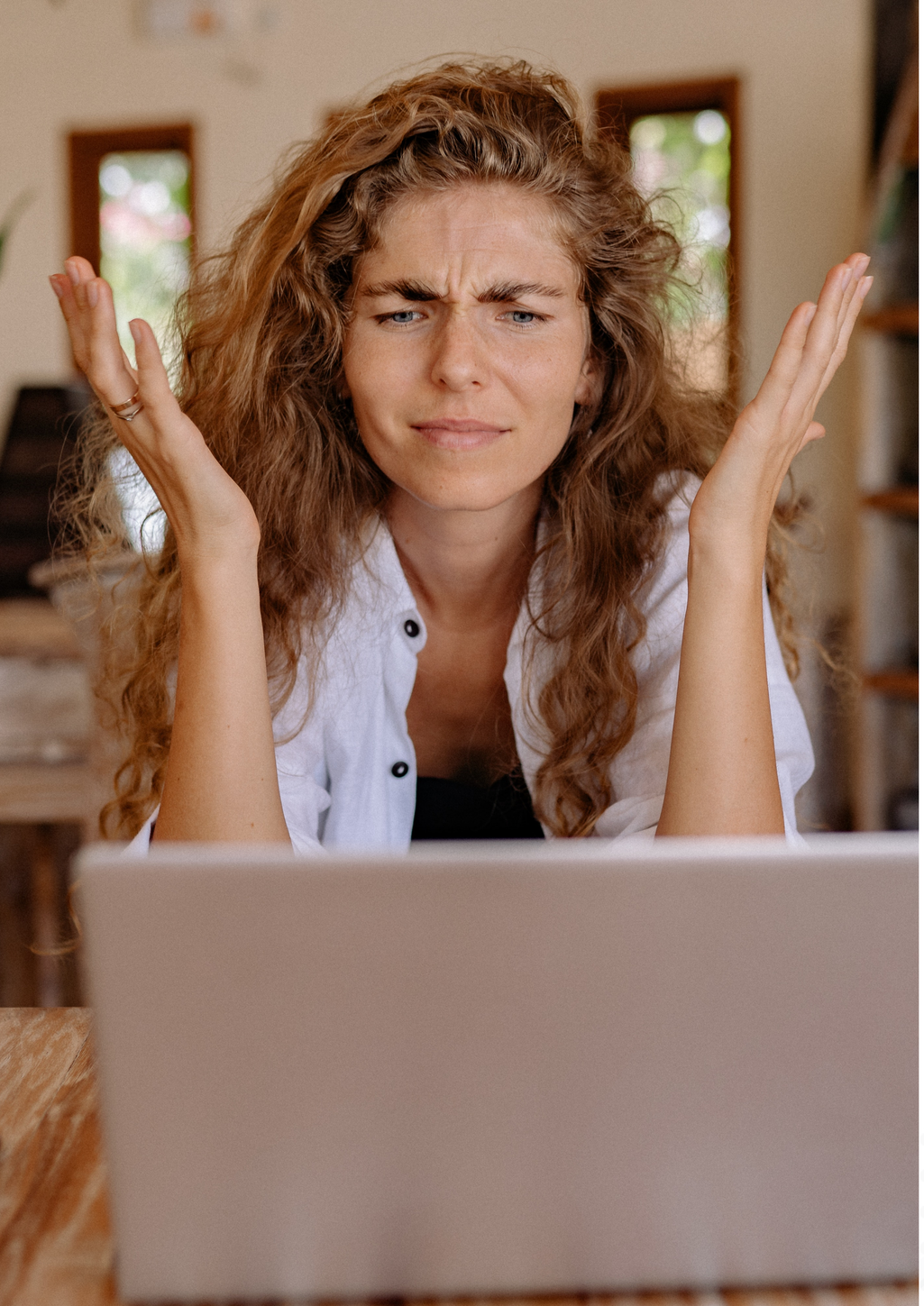
Improving Performance with Lazy Loading

Lara Newsom

Software Engineer | Speaker | Angular GDE

@laranerdsom



Large Initial Bundle Sizes

Adding features —> larger bundles

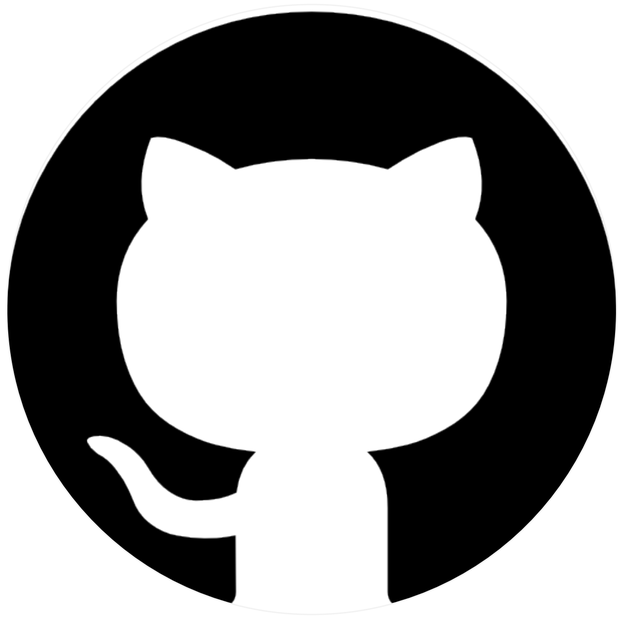
Larger bundles —> slower websites

Slow websites —> high bounce rates

High bounce rates —> missed profits

What We Will Cover in This Module

- Lazy loading for modules
- Lazy loading for standalone components
- Lazy loading route declarations
- Preloading strategies



Repository for Module Eight

github.com/lara-newsom/angular-routing-course

- module-eight-start

Angular Lazy Loading

Reduces initial bundle size

Also called code splitting

Splits the lazy loaded component or module off into a separate bundle

Lazy Loaded Code Organization

Separate code into separate directories

Use a monorepo tool like Nx to separate code into libraries

Makes it easier for teams to see code boundaries

The router understands code boundaries through the imports array of the lazy loaded component or module

Lazy loaded code is split off into separate javascript files

On initial page load the main javascript bundle is eagerly downloaded

Lazy loaded bundles are downloaded on the first visit to the route

How to Code Lazy Loaded Routes

Declare it in the route declaration

For modules or route arrays,
use loadChildren

For standalone components,
use loadComponent

Lazy loaded code can contain route declarations, router outlets, and initiate routing events

Comparing Bundle Sizes

Bundle size before lazy loading:

Initial Chunk Files	Names	Raw Size	Estimated Transfer Size
main.d0f1a25058febe42.js	main	525.71 kB	127.64 kB
styles.81f01cafd9f1f26a.css	styles	106.09 kB	9.63 kB
polyfills.a90d4e10f49a864c.js	polyfills	33.02 kB	10.64 kB
runtime.15ea7f952fde21ae.js	runtime	1.15 kB	628 bytes
	Initial Total	**665.97 kB**	**148.53 kB**

Comparing Bundle Sizes

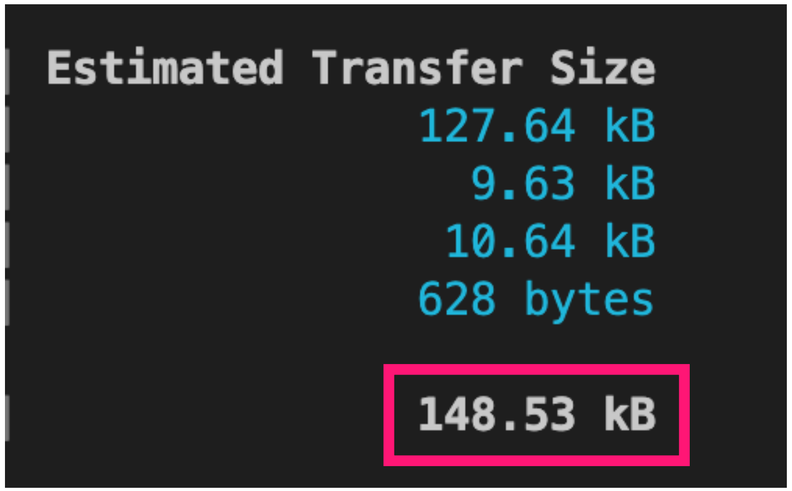
Bundle size after lazy loading:

Initial Chunk Files	Names	Raw Size	Estimated Transfer Size
main.0cce277d5ce25e11.js	main	498.51 kB	122.97 kB
styles.81f01cafd9f1f26a.css	styles	106.09 kB	9.63 kB
polyfills.a90d4e10f49a864c.js	polyfills	33.02 kB	10.64 kB
runtime.988d53b3a52220fb.js	runtime	2.72 kB	1.28 kB
	Initial Total	**640.33 kB**	**144.53 kB**

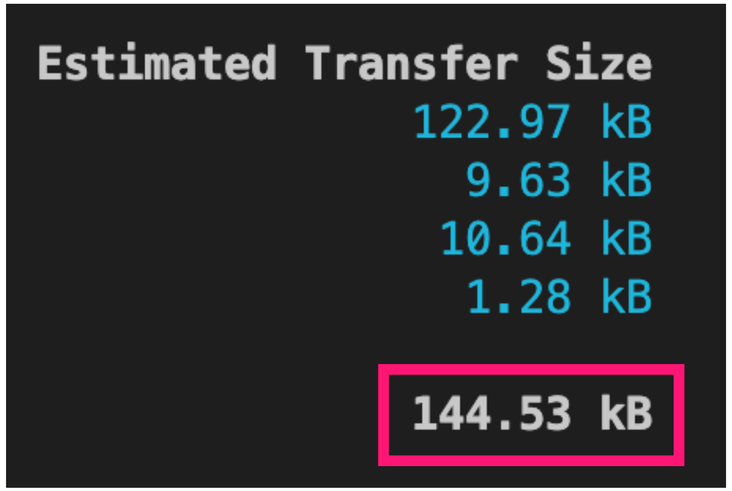
Lazy Chunk Files	Names	Raw Size	Estimated Transfer Size
205.99df71e77cd1dece.js	products-view-products-view-component	18.96 kB	4.96 kB
common.ce829df12e876847.js	common	9.39 kB	2.51 kB

Comparing Bundle Sizes

Before



After



Demo

Lazy loading a module based route

Demo

Lazy loading a standalone component route

Demo

Setting an application preloading strategy

Preloading lazy loaded route bundles improves user experience and reduces wait times

Preloading Strategy

Downloads the main bundle first

Then lazy loaded bundles are requested in the background

When users navigate to a lazy loaded route, the code is already downloaded

Set during router configuration

Configuring Module Based Preloading

```
@NgModule({
  imports: [
    RouterModule.forRoot(routes, {
      preloadingStrategy: PreloadAllModules
    })
  ],
```

Summary

- What lazy loading means
- How Angular separates the code
- How to declare a lazy loaded route
- Implemented a preloading strategy

Up Next:

Controlling User Navigation with Route Guards