Angular lazy loading example

angular lazyloading explain with example

Sure! Here’s a clear explanation of **Angular Lazy Loading** along with a simple example.

What is Angular Lazy Loading?

Lazy Loading in Angular is a technique to load **feature modules asynchronously** only when the user navigates to their routes, instead of loading all modules upfront when the application starts. This improves the app's initial load time and performance, especially for large apps.

Why use Lazy Loading?

- Reduce the **initial bundle size**.
- Load code **on demand** (when needed).
- Improve **user experience** by faster startup.
- Better **performance optimization**.

How Lazy Loading works in Angular?

- You split your app into feature modules.
- Each feature module is loaded only when its route is activated.
- Angular uses the `loadChildren` property in route configuration to load the module asynchronously.

Example: Angular Lazy Loading

Let's create a simple example with two modules:

- **AppModule** (main module)
- **AdminModule** (feature module loaded lazily)

Step 1: Create a Feature Module (`admin`)

In Angular CLI:

```bash
ng generate module admin --route admin --module app.module
```

This command:

- Creates `AdminModule` in `admin` folder.
- Sets up lazy loading route for path `'admin'`.
- Updates `app-routing.module.ts` with lazy loading.

Step 2: Understand the routing

After generation, your `app-routing.module.ts` will have something like:

```ts
const routes: Routes = [
 {
  path: 'admin',
  loadChildren: () => import('./admin/admin.module').then(m => m.AdminModule)
 }
];
```

This means:

- When user navigates to `/admin`, Angular dynamically loads the `AdminModule`.
- `AdminModule` itself has its own routes.

Step 3: Define routes inside `admin-routing.module.ts`

Example inside `admin-routing.module.ts`:

```ts
import { NgModule } from '@angular/core';
import { RouterModule, Routes } from '@angular/router';
import { AdminComponent } from './admin.component';
```

```
const routes: Routes = [
  { path: '', component: AdminComponent }
];

@NgModule({
  imports: [RouterModule.forChild(routes)],
  exports: [RouterModule]
})
export class AdminRoutingModule { }
```

Step 4: Create `AdminComponent`

Inside `admin` folder:

```
ts

import { Component } from '@angular/core';

@Component({
  selector: 'app-admin',
  template: `<h2>Admin Page Loaded Lazily!</h2>`
})
export class AdminComponent {}
```

Make sure it is declared in `AdminModule`.

Step 5: Run the app and test

- When you run `ng serve` and open the app, `/admin` module is NOT loaded initially.
- When you navigate to `http://localhost:4200/admin`, the AdminModule and AdminComponent load dynamically.
- You can verify via network tab in browser dev tools: you'll see the admin module chunk loaded only on demand.

Summary

- Use `loadChildren` in route config for lazy loading.

- Feature modules should have their own routing modules with `RouterModule.forChild()`.
- Improves initial load performance.
- Helps manage large Angular apps.

If you want, I can also provide a full minimal working example project structure and code! Would you like that?