Angular 2

Lesson 10—Routing

Learning Objectives

- Understand how Angular2 helps to achieve SPA using routing
- Define benefits of @NgModule
- Identify multiple ways of accessing routes
- Understand the process of routing lifecycle

Routing

Topic 1—Routers

What are Routers?

Angular2 Routers:

- Display the application component for the active URL.
- Manage navigation from one component to the next.
- Are often used in Single Page Applications (SPA).

Need for Routers

1. Routers segregate the application into different areas
2. Routers maintain the state in the application
3. Routers follow certain rules to protect areas of the app

The URL bar provides an edge to web applications over native applications.

- Routers permit you to reference states, bookmark them, and share them.
- In a good web application, an application state transition results in a URL change, and a modification in the URL alters the transition state. A URL is a serialized router state.

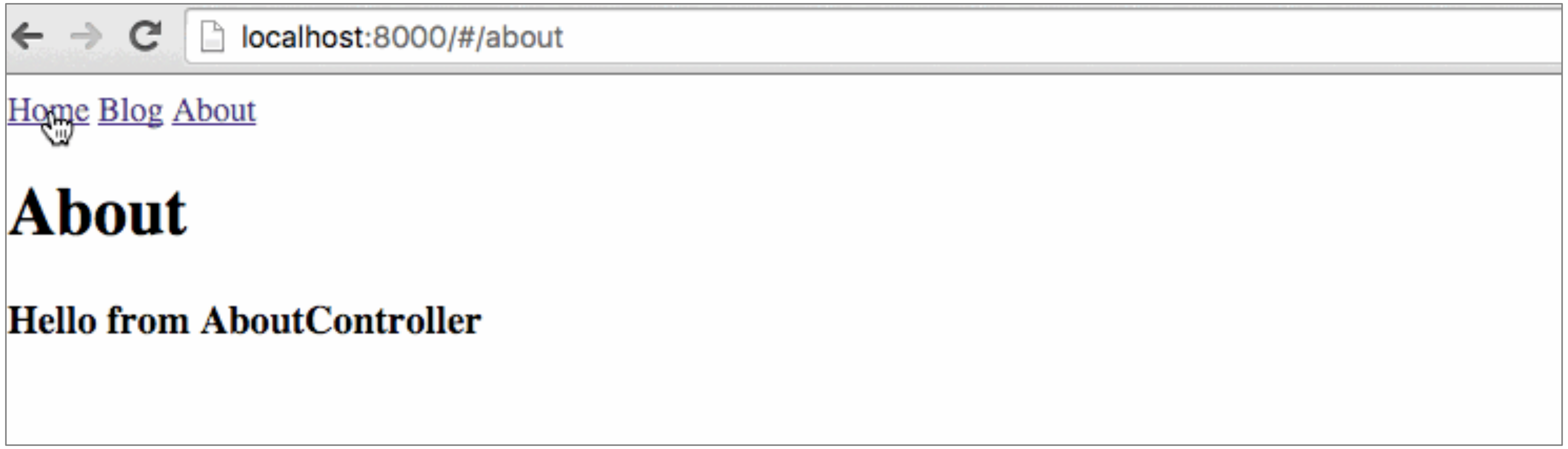
Routing

Topic 2—Single Page Application (SPA)

Single Page Application (SPA)

SPA is a single page web application where you can:

- Retrieve your code (CSS, HTML, JavaScript) with a single page load
- Navigate between pages without the need of refreshing the entire page


localhost:8000/#/about

Home Blog About

About

Hello from AboutController

Pros of Single Page Application (SPA)

No page refresh

There is no need to refresh the entire page. You can just load the part of the page that needs to be changed. Angular2 permits you to pre-load and cache all pages, so you don’t require extra requests to download them.

Improved user experience

SPAs act as native applications, which are fast and responsive.

Ability to work offline

With SPA, all pages are already loaded, so they work even if the internet connection is lost.

Cons of Single Page Application (SPA)

Complex to build

You are required to write JavaScript, handle shared state between pages, manage permissions, and more.

Slow initial load

SPA is slow in the initial load as it needs to download more resources when you open it.

JavaScript required

SPA requires clients to have JavaScript enabled.

Building SPA With Angular 2 Router

- Angular2 Router is an optional service that displays a distinct component view for a given URL.

- It is not a part of Angular2 core. It is in its own library package, @angular/router. You can import per your requirement as any other Angular2 package.

```
import { RouterModule, Routes } from '@angular/router';
```

Routing

Topic 3—NgModule

NgModule

NgModule is a way to organize dependencies for the compiler and dependency injection (DI).

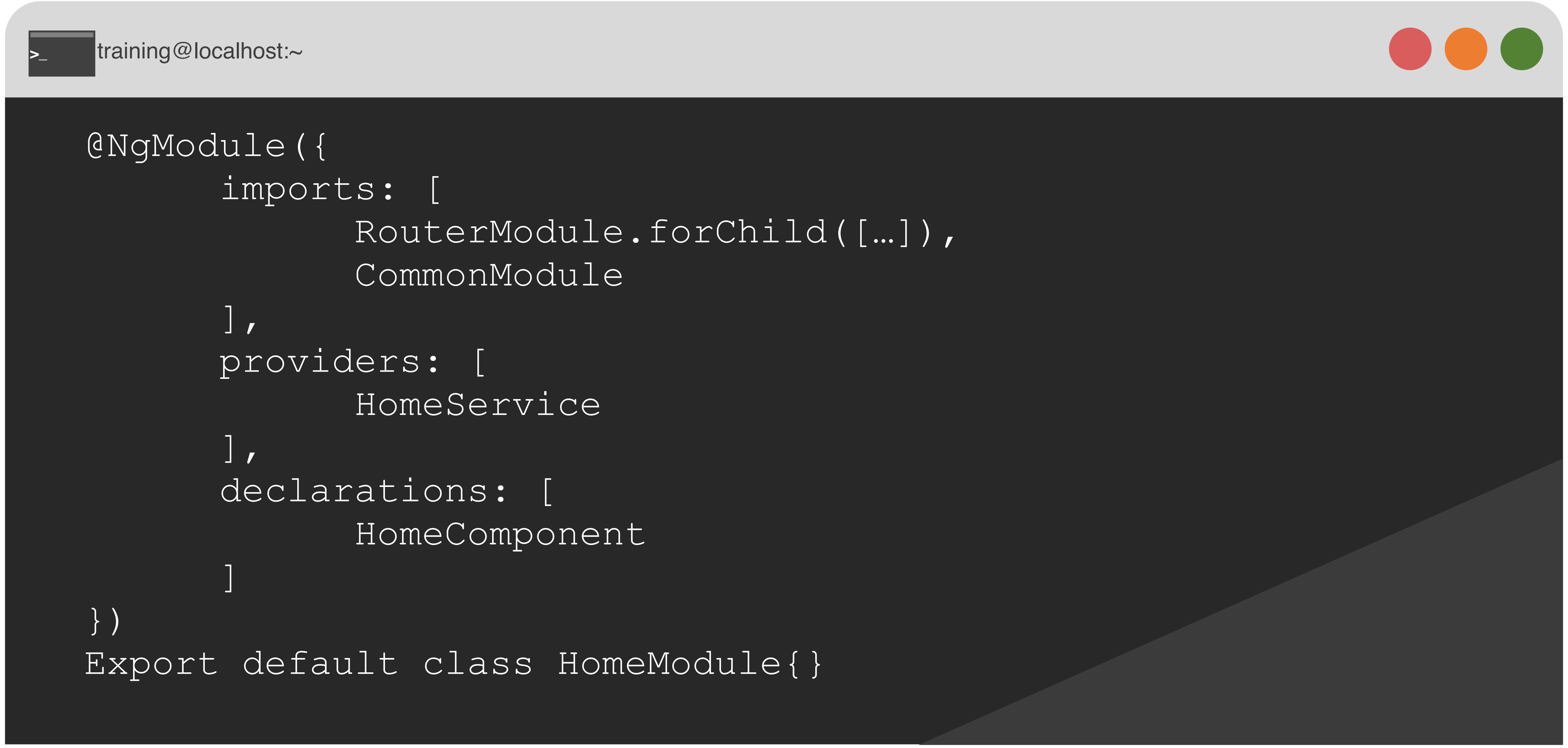
Main benefits of NgModule:

1. Dependency Injection
2. Lazy-load module with Router
3. Ahead of Time Compiler (AoT compiler)

NgModule

Module Feature

This is how the module file looks:



```
@NgModule({
      imports: [
            RouterModule.forChild([…]),
            CommonModule
      ],
      providers: [
            HomeService
      ],
      declarations: [
            HomeComponent
      ]
})
Export default class HomeModule{}
```

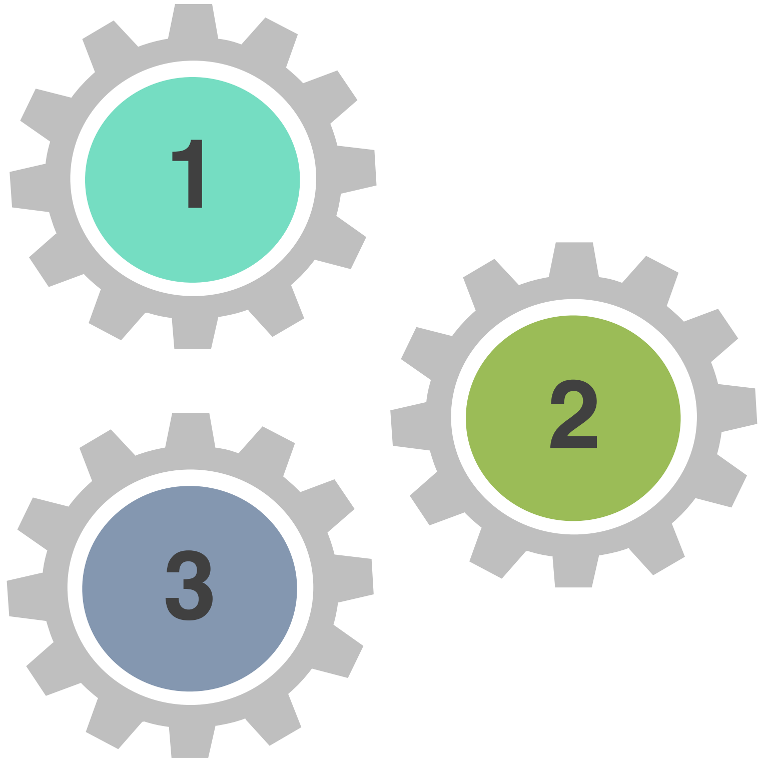
Routing

Topic 4—Router Basic

Router Basic

Main components used to configure routing:

Routers define the routes that your application supports

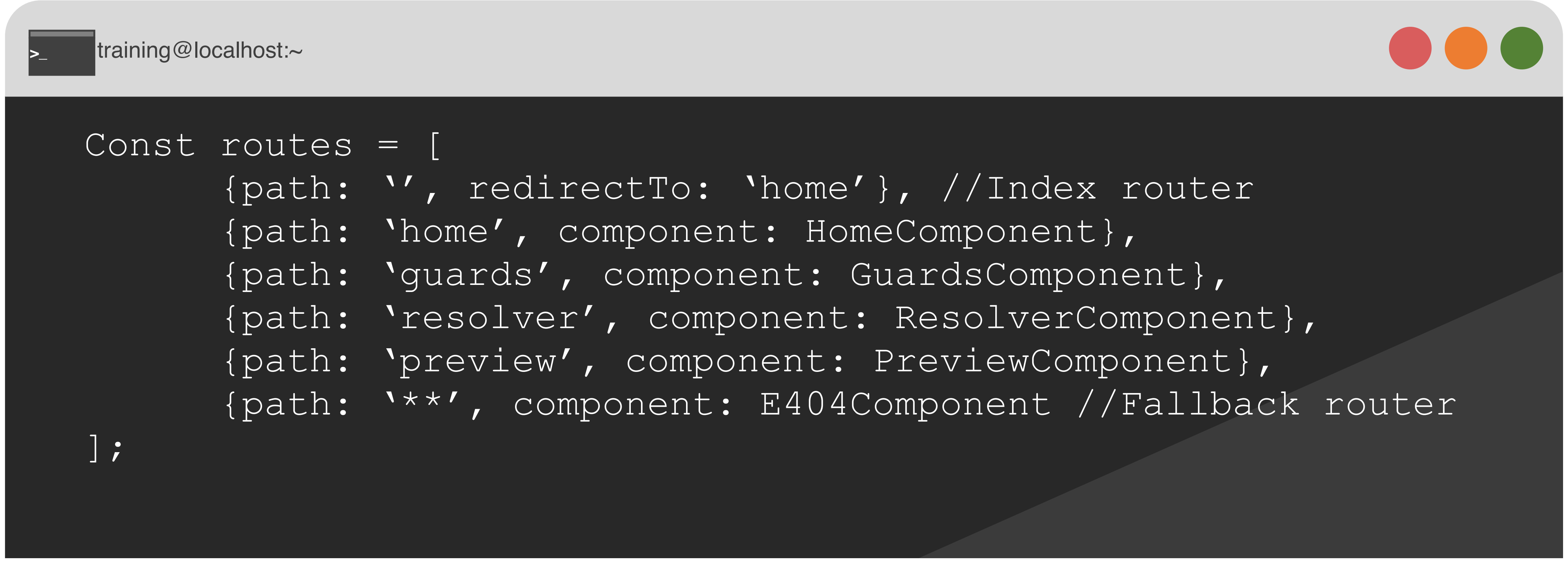


RouterOutlet is where the page content goes

RouterLink directive is used for navigation

Router Basic—Define Routes

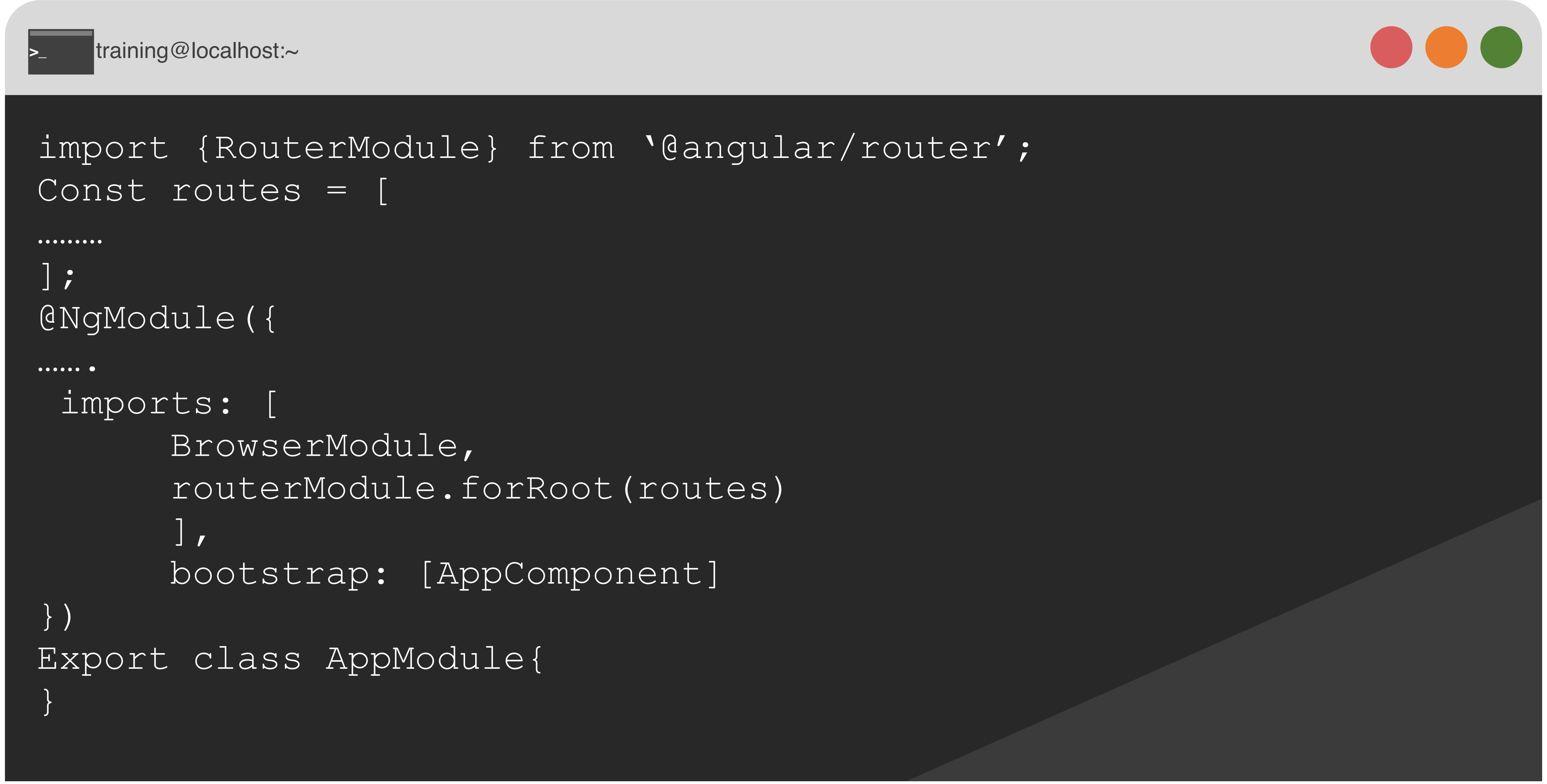
This is a mapping file where you can map components to their target URL.



```
Const routes = [
      {path: ‘’, redirectTo: ‘home’}, //Index router
      {path: ‘home’, component: HomeComponent},
      {path: ‘guards’, component: GuardsComponent},
      {path: ‘resolver’, component: ResolverComponent},
      {path: ‘preview’, component: PreviewComponent},
      {path: ‘**’, component: E404Component //Fallback router
];
```

Router Basic—Register Root Router

This is where the Root Router gets registered.

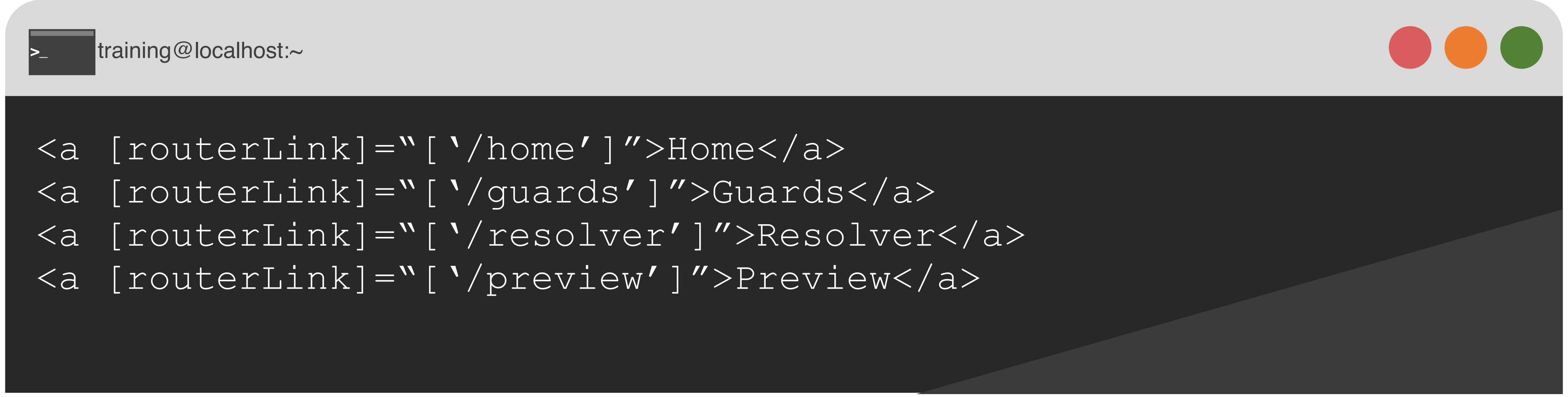


```
import {RouterModule} from ‘@angular/router’;
Const routes = [
………
];
@NgModule({
…….
 imports: [
      BrowserModule,
      routerModule.forRoot(routes)
      ],
      bootstrap: [AppComponent]
})
Export class AppModule{
}
```

Router Basic—Navigate to Link

routerLink Directive:

- routerLink directive is used to bind a clickable HTML element to a route.
- Click on an element with a routerLink directive that is bound to a string or a link parameters array, which triggers a navigation.

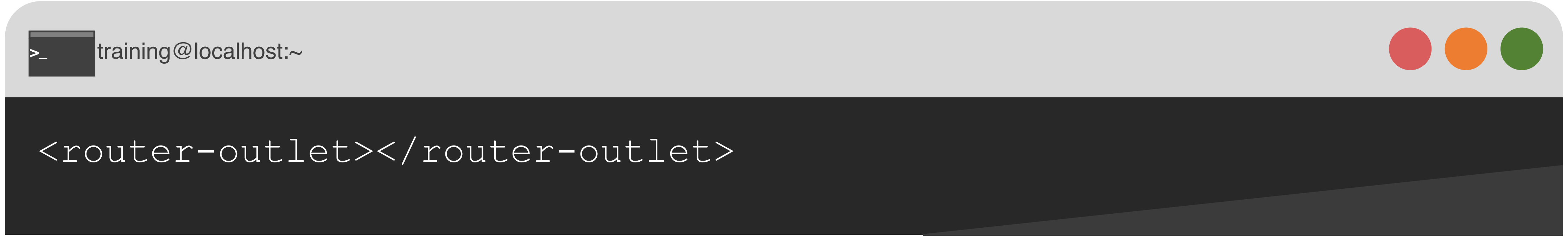


```
<a [routerLink]=“[‘/home’]”>Home</a>
<a [routerLink]=“[‘/guards’]”>Guards</a>
<a [routerLink]=“[‘/resolver’]”>Resolver</a>
<a [routerLink]=“[‘/preview’]”>Preview</a>
```

Router Basic—Rendering the Page

Router Outlet Directive:

- Marks where the router displays a view.
- Acts as a placeholder that Angular2 dynamically fills based on the current router state.



```
<router-outlet></router-outlet>
```

Demo—Basic Routing

DEMO

Routing

Topic 5—Parameter Routing

Parameter Routing

- In an application, you often want to navigate to a specific resource.
- For example, you have a news website with many articles. Every article may have an ID, and there is an article with ID 3.You may navigate to article 3 by visiting the URL:

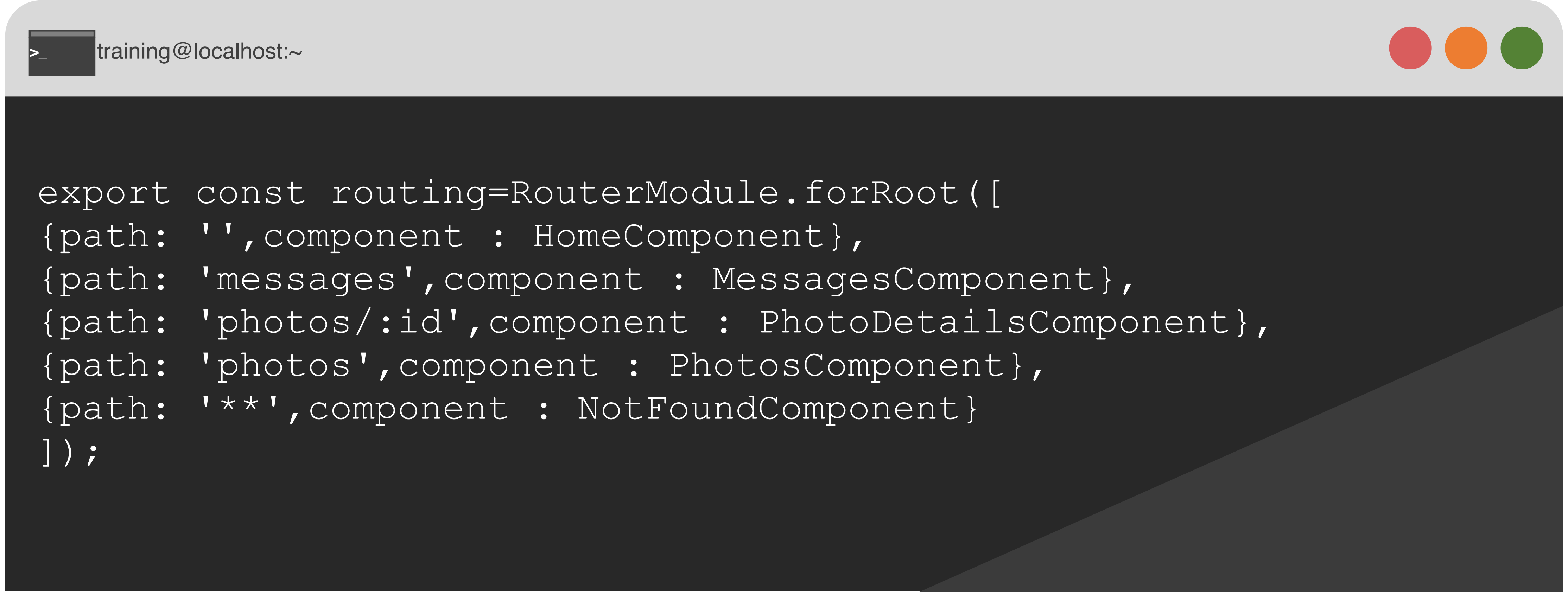
/articles/3

- Similarly, to access an article with ID 4, you may use the URL:

/articles/4

Parameter Routing

This is the way to indicate the route path by adding a parameter to router configuration:



```
export const routing=RouterModule.forRoot([
{path: '',component : HomeComponent},
{path: 'messages',component : MessagesComponent},
{path: 'photos/:id',component : PhotoDetailsComponent},
{path: 'photos',component : PhotosComponent},
{path: '**',component : NotFoundComponent}
]);
```

Demo—Parameter Routing

DEMO

Routing

Topic 6— Child Routes

Child Routes

Sometimes, when routes are accessible and viewed only within other routes, it is more relevant to create them as child routes.

Example :

- There is a tabbed navigations section in the product details page that presents the product overview by default.
- When the user clicks on the "Technical Specs" tab, the section shows the specs instead.

Declaration of Child Route

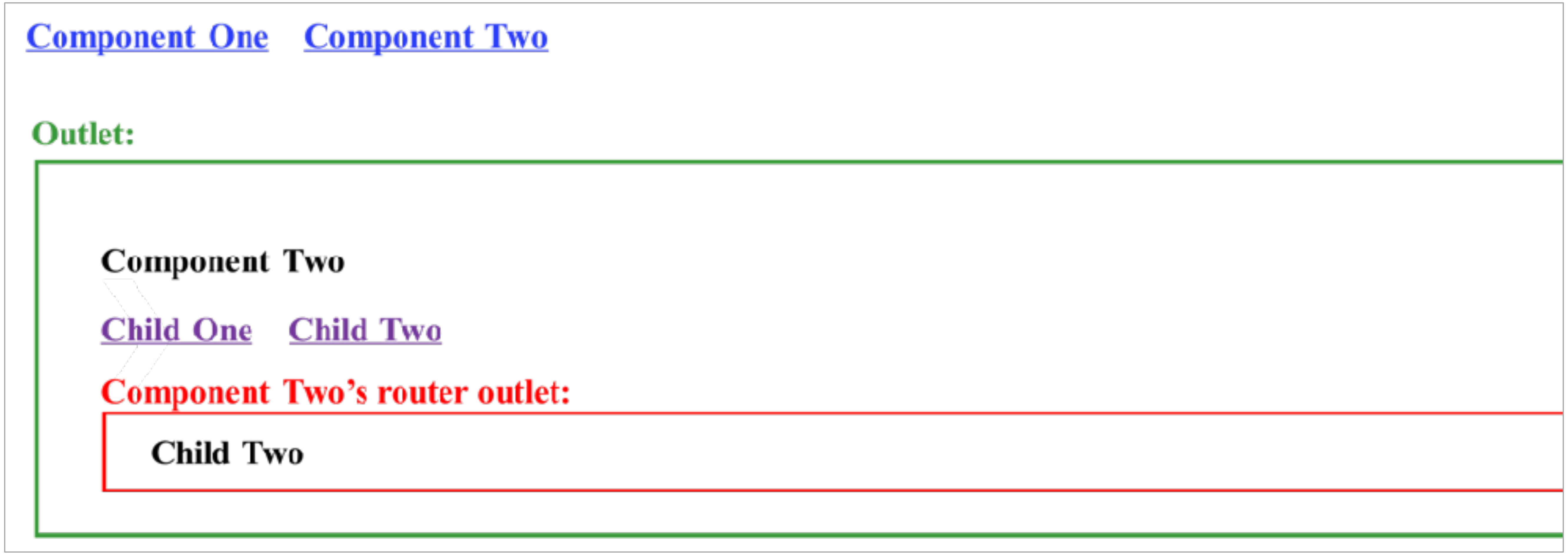
Example :

- In case the user clicks on the link of ID 3, show the user details page with the “overview”: localhost:3000/user-details/3/overview
- When the user clicks on “user skills": localhost:3000/user-details/3/skills
- Here, overview and skills are child routes of user-details/:id. They are reachable only within user details.

Child Routing Page

Example:

- In this page, both child one and two are child routes of component two.



Component One Component Two

Outlet:

Component Two

Child One Child Two

Component Two’s router outlet:

Child Two

Demo—Child Routing

DEMO

Routing

Topic 7—Router Lifecycle Hooks

Router Lifecycle Hooks

Sometimes, there is a need to carry out some action while changing routes.

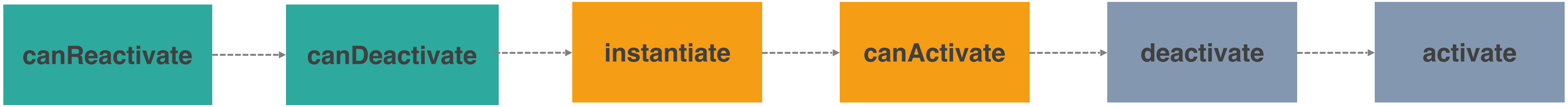
Actions:

A classical example of that is authentication.

- You have a login route and a protected route.
- You want to allow the app to go to the protected route only if the correct username and password are provided on the login page.
- To do that, you need to hook into the lifecycle of the router and ask to be notified when the protected route is being activated.
- You can then call an authentication service and ask whether or not the user provided the right credentials.

Router Lifecycle Hooks

Angular2 provides 6 different hooks :


canReactivate

canDeactivate

instantiate

canActivate

deactivate

activate

Router Lifecycle Hooks

The hooks are called in the illustrated sequence, from left to right.

On navigation from LoginComponent to ProtectedComponent routes, the sequence of triggers is:

- LoginComponent.canReactivate *if return is false, stops*;
- LoginComponent.canDeactivate *if return is false, stops*;
- ProtectedComponent.instantiate;
- ProtectedComponent.canActivate *if return is false, stops*;
- LoginComponent.deactivate;
- ProtectedComponent.activate;

Key Takeaways

- ✓ Router of Angular2 enables navigation from one view to the next as users perform application tasks.
- ✓ NgModule is a way to organize your dependencies for the compiler and Dependency injection (DI).
- ✓ Parameter routing and child routing are ways of accessing routers.
- ✓ Angular2 uses a lifecycle hook to carry out actions when changing routes.

Quiz

QUIZ

1

The ____________ directive substitutes the normal href property and makes it easier to work with route links in Angular2.

a. RouterLink

b. RouterRend

c. RouterLike

d. RouterLayer

QUIZ

1

The ____________ directive substitutes the normal href property and makes it easier to work with route links in Angular2.

a. RouterLink

b. RouterRend

c. RouterLike

d. RouterLayer

The correct answer is **a.**

The RouterLink directive substitutes the normal href property and makes it easier to work with route links in Angular2.

QUIZ

2

Which statement does NOT apply to routers?

a. Routers are used to control navigation

b. Routers are often used in Single Page Applications (SPA)

c. Routers are used to connect to back end services

d. Routers maintain the state in the application

QUIZ

2

Which statement does NOT apply to routers?

a. Routers are used to control navigation

b. Routers are often used in Single Page Applications (SPA)

c. Routers are used to connect to back end services

d. Routers maintain the state in the application

The correct answer is **c.**

Routers are not used to connect to back end services.

QUIZ

3

The router in Angular2 has been reworked to be simple, yet extensible. It will include the following basic features:

a. Simple JSON-based Route Config

b. Optional Convention over Configuration

c. Static, Parameterized, and Splat Route Patterns

d. URL Unresolver

e. All of the above

QUIZ

3

The router in Angular2 has been reworked to be simple, yet extensible. It will include the following basic features:

a. Simple JSON-based Route Config

b. Optional Convention over Configuration

c. Static, Parameterized, and Splat Route Patterns

d. URL Unresolver

e. All of the above

The correct answer is **a, b, and c**

The router in Angular2 is simple, yet extensible. It includes simple JSON-based Routing Config, optional Convention over Configuration and Static, Parameterized, and Splat Route Patterns.

Thank You