How to apply canActivate guard on all the routes?

49

12

I have a angular2 active guard which handle if the user is not logged in, redirect it to login page:

```
import { Injectable } from  "@angular/core";
import { CanActivate , ActivatedRouteSnapshot, RouterStateSnapshot, Router} from
"@angular/router";
import {Observable} from "rxjs";
import {TokenService} from "./token.service";

@Injectable()
export class AuthenticationGuard implements CanActivate {

    constructor (
        private router : Router,
        private token : TokenService
    ) { }

    /**
     * Check if the user is logged in before calling http
     *
     * @param route
     * @param state
     * @returns {boolean}
     */
    canActivate (
        route : ActivatedRouteSnapshot,
        state : RouterStateSnapshot
    ): Observable<boolean> | Promise<boolean> | boolean {
        if(this.token.isLoggedIn()){
            return true;
        }
        this.router.navigate(['/login'],{ queryParams: { returnUrl: state.url }});
        return;
    }
}
```

```
    { path : ':id', component: UserShowComponent },
    { path : 'delete/:id', component : DeleteComponent, canActivate:
[AuthenticationGuard] },
    { path : 'ban/:id', component : BanComponent, canActivate:[AuthenticationGuard] },
    { path : 'edit/:id', component : EditComponent, canActivate:[AuthenticationGuard] }
];
```

Is there any better way to implement canActive option without adding it to each path.

What I want is to add it on main route, and it should apply to all other routes. I have searched alot, but I could not find any useful solution

Thanks

angular typescript angular-routing

edited Apr 19 '17 at 6:23
Rahul Kumar
3,200 4 27 39

asked Apr 19 '17 at 6:16
Nimatullah Razmjo
556 1 7 27

4 Answers

114

✓

You can introduce a componentless parent route and apply the guard there:

```
const routes: Routes = [
    {path: '', canActivate:[AuthenticationGuard], children: [
      { path : '', component: UsersListComponent },
      { path : 'add', component : AddComponent},
      { path : ':id', component: UserShowComponent },
      { path : 'delete/:id', component : DeleteComponent },
      { path : 'ban/:id', component : BanComponent },
      { path : 'edit/:id', component : EditComponent }
    ]}
```

353k 81 1110 1017

Thanks, It works by now – Nimatullah Razmjo Apr 19 '17 at 6:43

Glad to hear :) Thanks for the feedback. – Günter Zöchbauer Apr 19 '17 at 6:44

7 @GünterZöchbauer I always read your advice. Thank you very much!! But in my case this does not work. CanActive does not work in routing between children. I use canActivateChild. It works. – Smiranin Feb 13 '18 at 14:06

2 I'm not sure. You could try `canActivateChild` instead of `canActivate` . Haven't used that myself yet. – Günter Zöchbauer Apr 20 '18 at 2:45

2 Yup, `canActivateChild` is a better approach for the given scenario to envelope all child routes. And the parent itself can guarded using `canActivate` or `canLoad` depending on when its loaded in the app. Also, the code snippet will give console errors in v6.0 as Angular will expect a component or child declaration in the route config. – ashish.gd Sep 27 '18 at 18:48

6

You can also subscribe to the router's route changes in your app.component's ngOnInit function and check authentication from there e.g.

```
this.router.events.subscribe(event => {
    if (event instanceof NavigationStart && !this.token.isLoggedIn()) {
        this.router.navigate(['/login'],{ queryParams: { returnUrl: state.url}});
    }
});
```

I prefer this way of doing any kind of app wide check(s) when a route changes.

answered Feb 22 '18 at 11:45
Alan Smith
463 2 14 20

I think you should implement "child routing" which allow you to have a parent (with a path "admin" for example) and his childs.

Then you can apply a canactivate to the parent which will automatically restrict the access to all his child. For example if I want to a 67 6

-2

I came across this example when searching and was caught out by the example given in the example.

You need to ensure that the guard returns true if you want to show the child routes.

```
@Injectable()
export class AuthenticationGuard implements CanActivate {

    constructor(
        private router: Router,
        private authService: AuthService) { }

    canActivate(
        route: ActivatedRouteSnapshot,
        state: RouterStateSnapshot
    ): Observable<boolean> | Promise<boolean> | boolean {

        // Auth checking code here

        // Make sure you return true here if you want to show child routes
        return true;
    }
}
```

answered Feb 21 '18 at 13:03

CountZero
3,660 3 36 50