Jonathan Roland

Web 425 Angular with TypeScript

Prof. Richard Krasso

1/14/2021

Discussion 3.1 – Guarding Routes

Angular routes define the navigation of the user interface for our users, but if a user wants to enter or leave a route, we must set up some form of validation. There are many reasons we might want to do validation, such as authentication, gated navigation (needing to fill out form A before form B), or to stop the user from losing unsaved changes.

Conventionally, we can use the canActive() and canDeactivate() functions to set up route guards. The functions expect Boolean true or false from custom functions provided by the developer to tell Angular whether the user can activate or de-activate a route. To use these route guard functions, we must first create a custom class that extends one of them and hook up our validation business logic to this class. Then we will need to add this class to the router configuration for the components we want it applied to.

We can also use route guarding for purposes other than authentication. Sometimes we don’t want the user to enter a route if it has not finished loading data. For this we can use the resolve() method. The setup is similar to canActivate() but will instead await confirmation that data loading has resolved before allowing the user to enter or exit the route. We may also wish to discern whether child routes can be loaded using the canLoad() method. This method either returns false, stopping navigation, or returns a UrlTree object, which starts a new navigation.

It is important to keep in mind that these are not true security features since they run on the front end and can be altered. The purpose of route guarding is to keep the user within the flow we intend for them so we can provide an optimal user experience.

**References:**

Fain, Y., & Moiseev, A. (2016). *Angular 2 Development with TypeScript* (1st ed.). Manning Publications.

*Angular*. (n.d.-b). Angular. Retrieved January 14, 2021, from https://angular.io/guide/router#preventing-unauthorized-access