

Der Angular Router

- SinglePageApplication → Links führen zu DOM-Änderung ohne Postback zum Server
 - Sonst wäre Status am Client zerstört
- Angular „fängt“ die Navigation ab
- Router ist weiteres Angular-Modul
- Router ist konfigurierbar
 - Route → Component
 - Dynamische Routen sind möglich (.../id)
 - Querystringparameter werden unterstützt

RouterModule einbinden app.module

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

- Routen definieren
- Reihenfolge ist wichtig
- Parameter sind möglich
- `**,` „fängt“ restliche Routen auf

```
RouterModule.forRoot(  
  [  
    { path : '', component : HomeComponent},  
    { path : 'followers/:id', component : GithubProfileComponent },  
    { path : 'followers', component : GithubFollowersComponent},  
    { path : 'posts', component : PostsComponent },  
    { path : '**', component : NotFoundComponent }  
  ]  
)
```

Components noch ohne Funktion

- Nur PostsComponent
 - Wie gehabt
- GitHubFollowers
 - Lädt Followers von Mosh Hamedani
 - Nach Muster Posts

```
home.component.html x
1 <p>
2   home works!
3 </p>
```

```
home.component.ts x
1 import { Component, OnInit } from '@angular/core';
2
3 @Component({
4   selector: 'app-home',
5   templateUrl: './home.component.html',
6   styleUrls: ['./home.component.css']
7 })
8 export class HomeComponent implements OnInit {
9
10  constructor() { }
11
12  ngOnInit() {
13  }
14
15 }
```

GitHubFollowers Service

github-followers.service.ts x

```
1  import { Injectable, OnInit } from '@angular/core';
2  import { HttpClient } from '@angular/common/http';
3  import { DataService } from '../data.service';
4  import { GithubFollowersComponent } from '../github-followers/github-followers.component';
5
6  @Injectable()
7  export class GithubFollowersService extends DataService<Follower> {
8
9      constructor(http: HttpClient) {
10         super('https://api.github.com/users/mosh-hamedani/followers?per_page=5', http);
11     }
12 }
```

GitHubFollowers - Template

github-followers.component.html x


```
1 <div *ngFor="let follower of followers" class="media">
2   <div class="media-left">
3     <a href="#">
4       
6   </div>
7   <div class="media-body">
8     <h4 class="media-heading">
9       <a href="#">{{ follower.login }}</a>
10    </h4>
11    <a href="{{ follower.html_url }}">{{ follower.html_url }}</a>
12  </div>
13 </div>
```

github-followers.component.css x


```
1
2 .avatar {
3   width: 80px;
4   height: 80px;
5   border-radius: 100%;
6 }
```

localhost:4200/followers


Followers Posts




SeaBassTian
<https://github.com/SeaBassTian>




lfurzewaddock
<https://github.com/lfurzewaddock>



OMENSAH
<https://github.com/OMENSAH>



yusijs
<https://github.com/yusijs>



kiuka
<https://github.com/kiuka>

GitHubFollowersComponent – wie gehabt

github-followers.component.ts ✕

```
1  import { GithubFollowersService } from '../services/github-followers.service';
2  import { Component, OnInit } from '@angular/core';
3
4  @Component({
5    selector: 'github-followers',
6    templateUrl: './github-followers.component.html',
7    styleUrls: ['./github-followers.component.css']
8  })
9  export class GithubFollowersComponent implements OnInit {
10    followers: Follower[];
11
12    constructor(private service: GithubFollowersService) { }
13
14    ngOnInit() {
15      this.service.get()
16        .subscribe(followers => this.followers = followers);
17    }
18  }
```

Routen verlinken

- Attribut href führt zu Postbacks
- Keine SPA
- Directive router-link verwenden

navbar.component.html x

```
1 <nav class="navbar navbar-default">
2   <div class="container-fluid">
3     <div class="collapse navbar-collapse">
4       <ul class="nav navbar-nav">
5         <li class="active"><a href="/followers">Followers</a></li>
6         <li><a href="/posts">Posts</a></li>
7       </ul>
8     </div>
9   </div>
10 </nav>
```

Dynamischen Link verwenden

- PropertyBinding auf routerLink
 - Route wird als Array übergeben
 - Erstes Element → path
 - Weitere Elemente → Parameter

github-followers.component.html x

```
1 <div *ngFor="let follower of followers" class="media">
2   <div class="media-left">
3     <a href="#">
4       
5     </a>
6   </div>
7   <div class="media-body">
8     <h4 class="media-heading">
9       <a [routerLink]="['followers', follower.id]">{{ follower.login }}</a>
10    </h4>
11    <a href="follower.html_url">{{ follower.html_url }}</a>
12  </div>
13</div>
```


Aktiven Link hervorheben → routerLinkActive

- PropertyBinding funktioniert nicht mehr

```
<ul class="nav navbar-nav">  
  <li routerLinkActive="active current"><a routerLink="/followers">Followers</a></li>  
  <li routerLinkActive="active current"><a routerLink="/posts">Posts</a></li>  
</ul>
```

- Wenn eine Route die andere enthält → Options

```
[routerLinkActiveOptions="{exact: true}"]
```

Ergebnis

← → ↻ 🏠 ⓘ localhost:4200/posts

Followers

Posts

UpdatePatchDelete

Angular in der Htl
leonding

UpdatePatchDelete

C#

UpdatePatchDelete

Nach Update

Routingparameter programmatisch auslesen

- Häufig wird id übergeben und Daten für die id sind zu laden
- Params sind in onInit() auszulesen
- Wenn Komponente nicht neu erzeugt wird (z.B. Aufruf aus selber Component) geht neue id verloren
 - Observable für Parameter → subscribe()
 - Wenn Fall nie auftritt → statische Übernahme möglich

Quelle: Liste der Follower

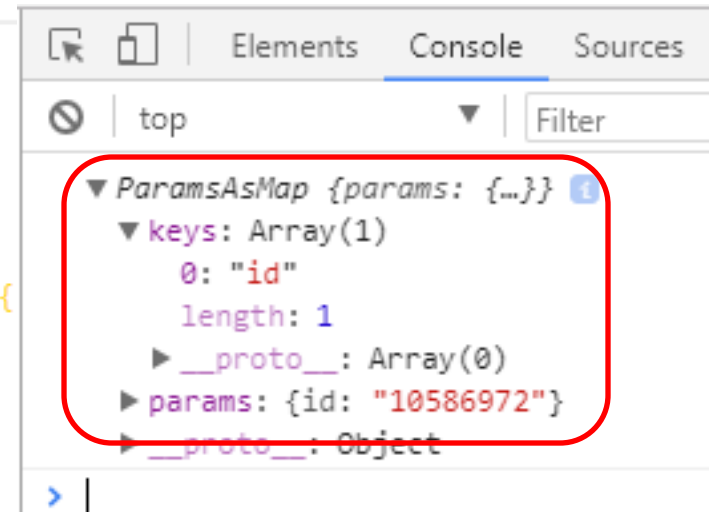
- Parameter: id

```
github-followers.component.html ●  
1  <div *ngFor="let follower of followers" class="media">  
2    <div class="media-left">  
3      <a href="#">  
4          
6      </a>  
7    </div>  
8    <div class="media-body">  
9      <h4 class="media-heading">  
10       <a [routerLink]="['/followers', follower.id]">  
11         {{ follower.login }}</a>  
12      </h4>  
13      <a href="follower.html_url">{{ follower.html_url }}</a>  
14    </div>  
15  </div>
```

Ziel: Detailseite

github-profile.component.ts

```
1 import { Component, OnInit } from '@angular/core';
2 import { ActivatedRoute } from '@angular/router';
3
4 @Component({
5   selector: 'app-github-profile',
6   templateUrl: './github-profile.component.html',
7   styleUrls: ['./github-profile.component.css']
8 })
9 export class GithubProfileComponent implements OnInit {
10
11   constructor(private route : ActivatedRoute) { }
12
13   ngOnInit() {
14     this.route.paramMap
15       .subscribe ( params =>
16         {
17           console.log(params)
18         }
19       );
20   }
21 }
```



Routen auslagern app.routing.ts

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

import { HomeComponent } from "../home.component";
import { LoginComponent } from "../login.component";
import { RegisterComponent } from "../register.component";
import { UserComponent } from "../user/user.component";

const APP_ROUTES: Routes = [
  { path: '', component: HomeComponent },
  { path: 'user', component: UserComponent }
];

export const routing = RouterModule.forRoot(APP_ROUTES);
```

In app.module einbinden

```
import { routing } from "../app.routing";
```

```
@NgModule({  
  declarations: [  
    AppComponent,  
    UserComponent,  
    UserDetailComponent,  
    UserEditComponent,  
    HomeComponent  
  ],  
  imports: [  
    BrowserModule,  
    routing  
  ],  
})
```

- Rootelement dynamisch setzen

```
<div class="container">  
  <h1>Routing</h1>  
  <router-outlet></router-outlet>  
</div>
```


RouterLinks relativ setzen

```
<div class="container">
  <div class="row">
    <div class="col-xs-12">
      <a [routerLink]="['']">Home</a>
      <a [routerLink]="['user']">User</a>
    </div>
  </div>
  <router-outlet></router-outlet>
</div>
```

Links absolut setzen

- Auch ../user ist möglich

```
<div class="container">  
  <div class="row">  
    <div class="col-xs-12">  
      <a [routerLink]="['/']">Home</a>  
      <a [routerLink]="['/user']">User</a>  
    </div>  
  </div>  
  <router-outlet></router-outlet>  
</div>
```

Programmatisch navigieren

```
import { Component } from "@angular/core";
import { Router } from "@angular/router";

@Component({
  template: `
    <div class="row">
      <div class="col-xs-12">
        <h2>Dein Account</h2>
        <button class="btn btn-primary" (click)="onNavigate()">Zur Startseite</button>
      </div>
    </div>
  `
})
export class UserComponent {
  constructor(private router: Router) {}

  onNavigate() {
    this.router.navigate(['/']);
  }
}
```

Routingparameter verwenden

- Z.B. id bei Resource

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

import { UserComponent } from "../user/user.component";
import { HomeComponent } from "../home.component";

const APP_ROUTES: Routes = [
  { path: '', component: HomeComponent },
  { path: 'user/details' },
  { path: 'user/:id', component: UserComponent }
];

export const routing = RouterModule.forRoot(APP_ROUTES);
```

Routingparameter extrahieren

- Ideal über Subscription der Parameteränderungen
 - DI von ActivatedRoute

```
export class UserComponent implements OnInit {  
  id: string;  
  
  constructor(private router: Router, private activatedRoute: ActivatedRoute) {}  
  
  onNavigate() {  
    this.router.navigate(['/']);  
  }  
  
  ngOnInit() {  
    // this.id = this.activatedRoute.snapshot.params['id']  
    this.activatedRoute.params.subscribe(  
      (params: Params) => this.id = params['id']  
    );  
  }  
}
```

Thema unsubscribe

- Prinzipiell muss jedes **.subscribe** mit einem **.unsubscribe** wieder freigegeben werden.
- Bei Routingparameter wird dies jedoch (genauso wie bei den Methode den HttpModule) **automatisch** durchgeführt

When subscribing to an observable in a component, you almost always arrange to unsubscribe when the component is destroyed.

There are a few exceptional observables where this is not necessary. The `ActivatedRoute` observables are among the exceptions.

The `ActivatedRoute` and its observables are insulated from the Router itself. The Router destroys a routed component when it is no longer needed and the injected `ActivatedRoute` dies with it.

Feel free to unsubscribe anyway. It is harmless and never a bad practice.