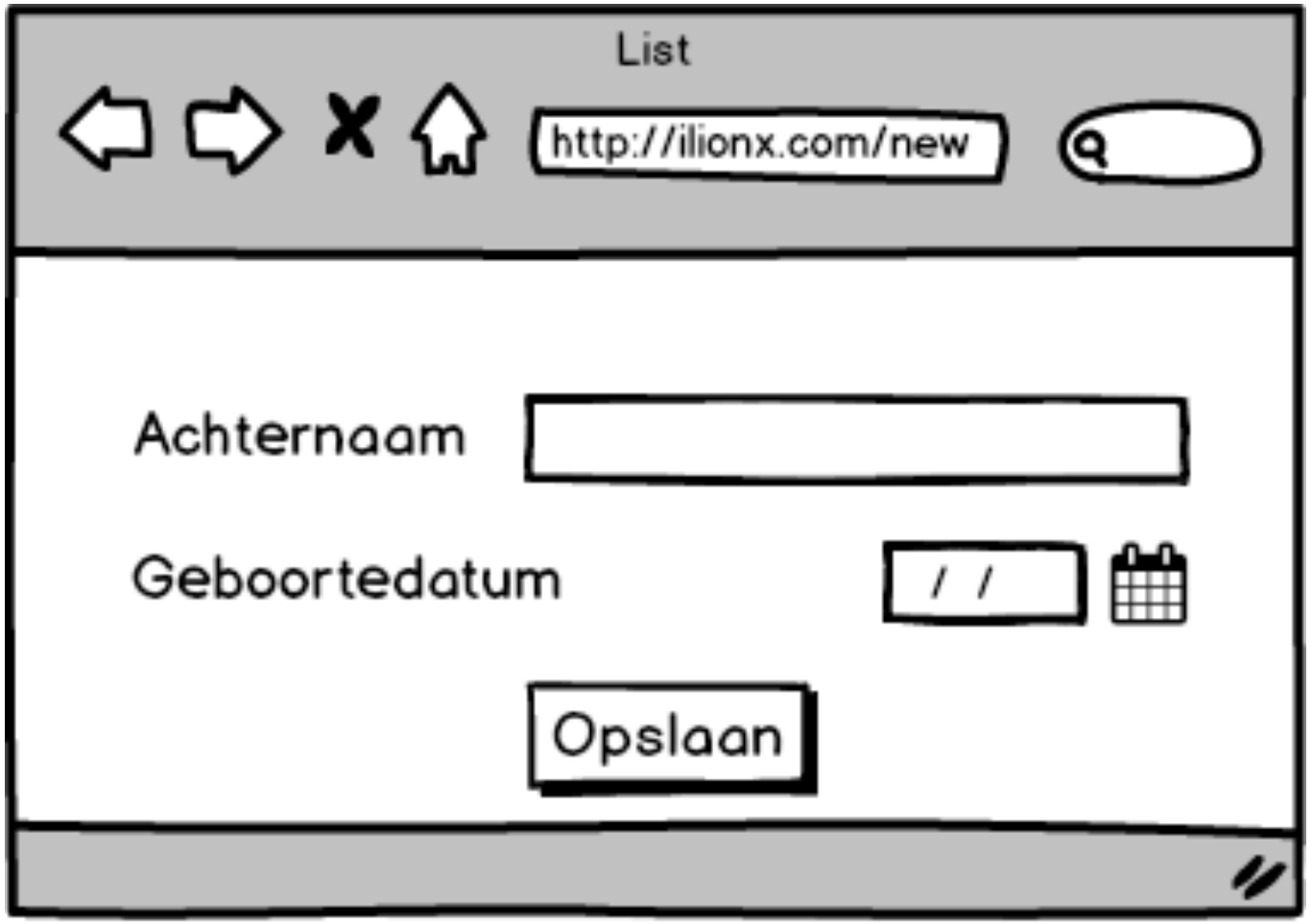
Routing

Routing

- Divide the application into separate blocks
- Decides the "active tree"
- Securing urls
- `RouterModule`

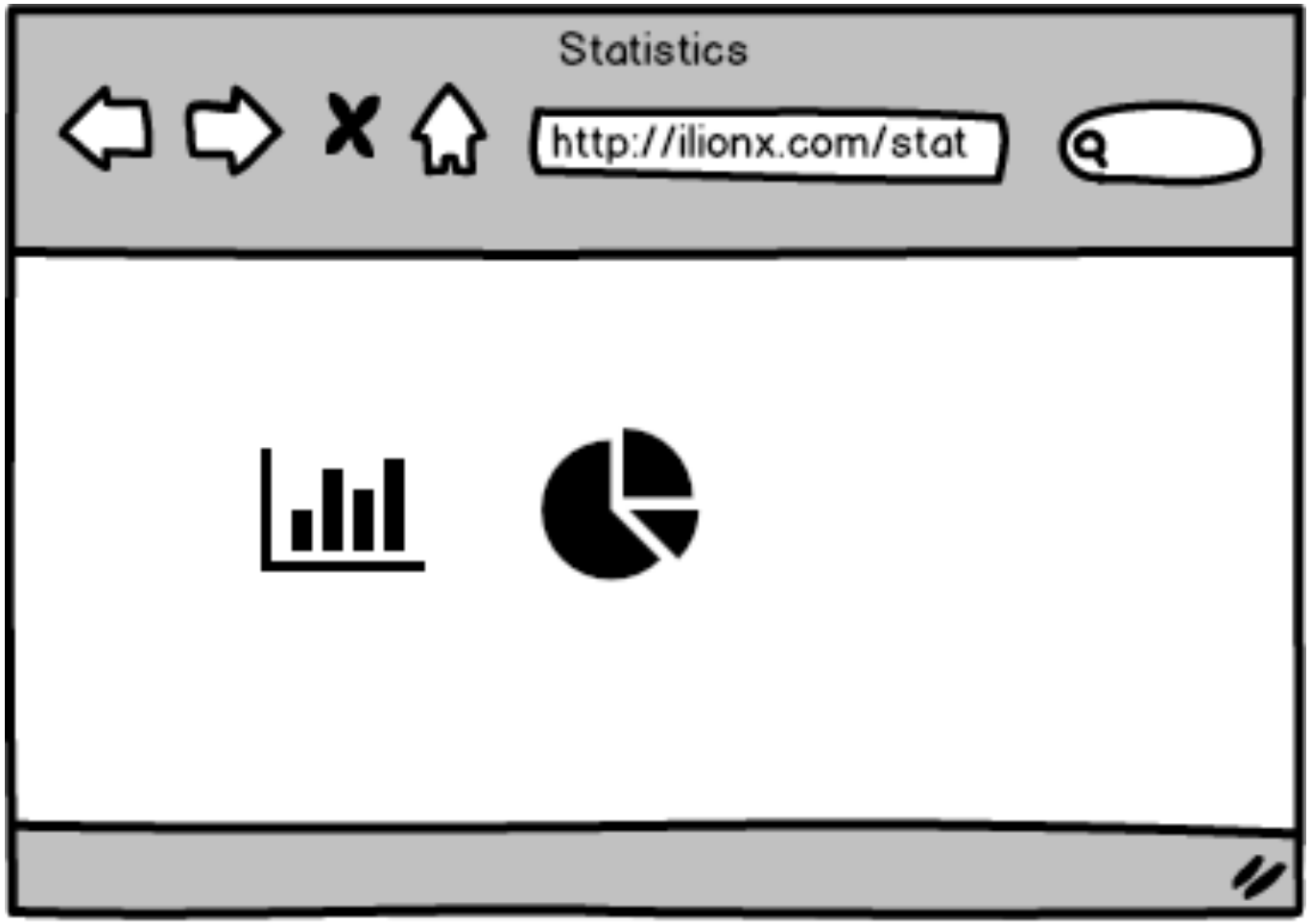
Routing



ilionx
Your partner in digital business

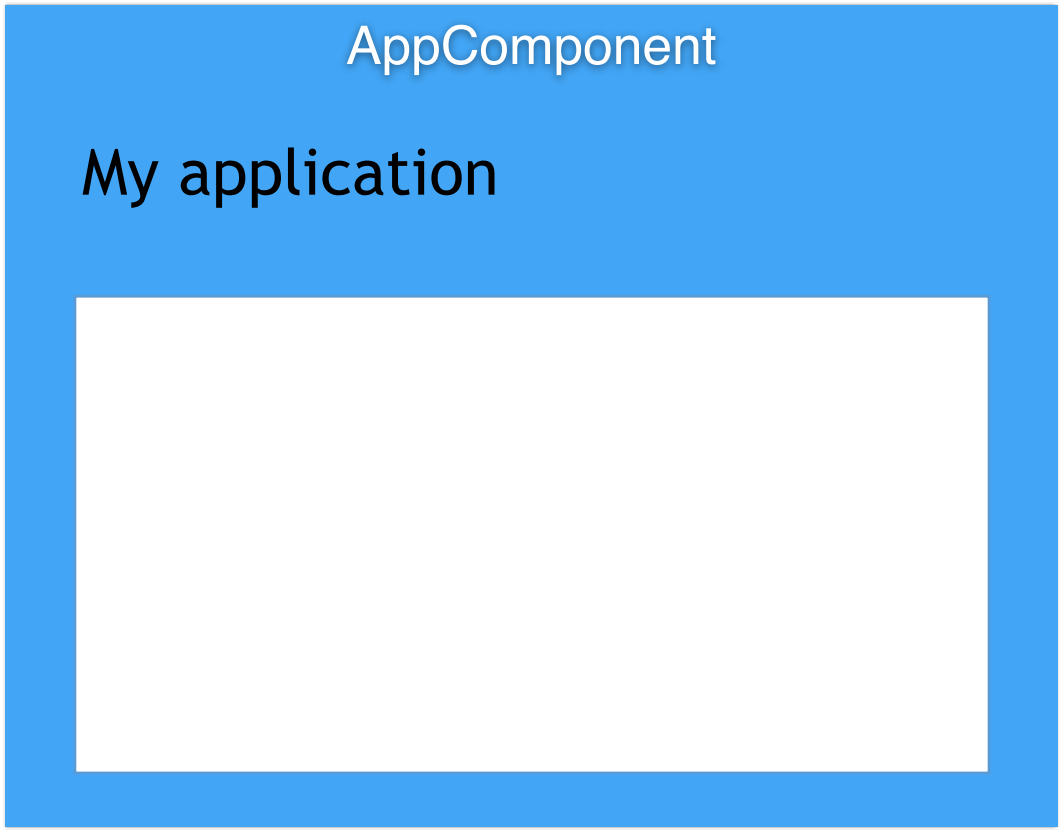
Routing

www.ilionx.com/statistics



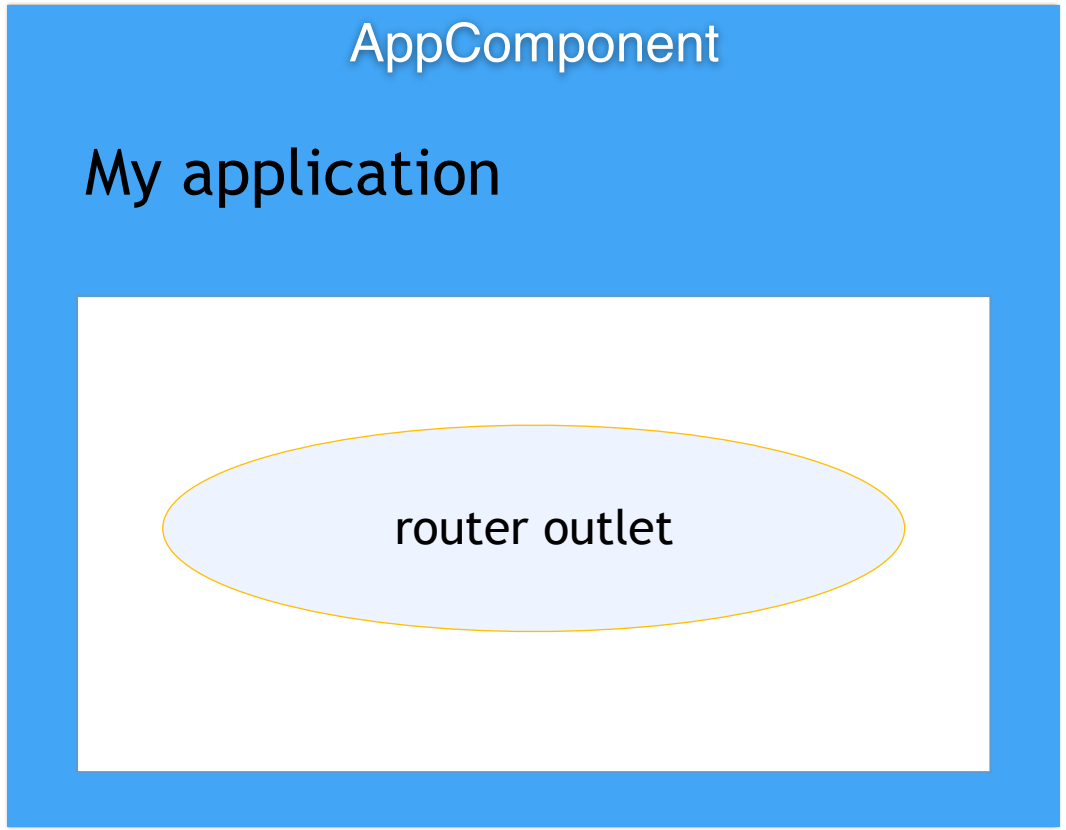
ilionx
Your partner in digital business

Router outlet




```
<div id="app-component">

  <h1>My application</h1>



</div>
```

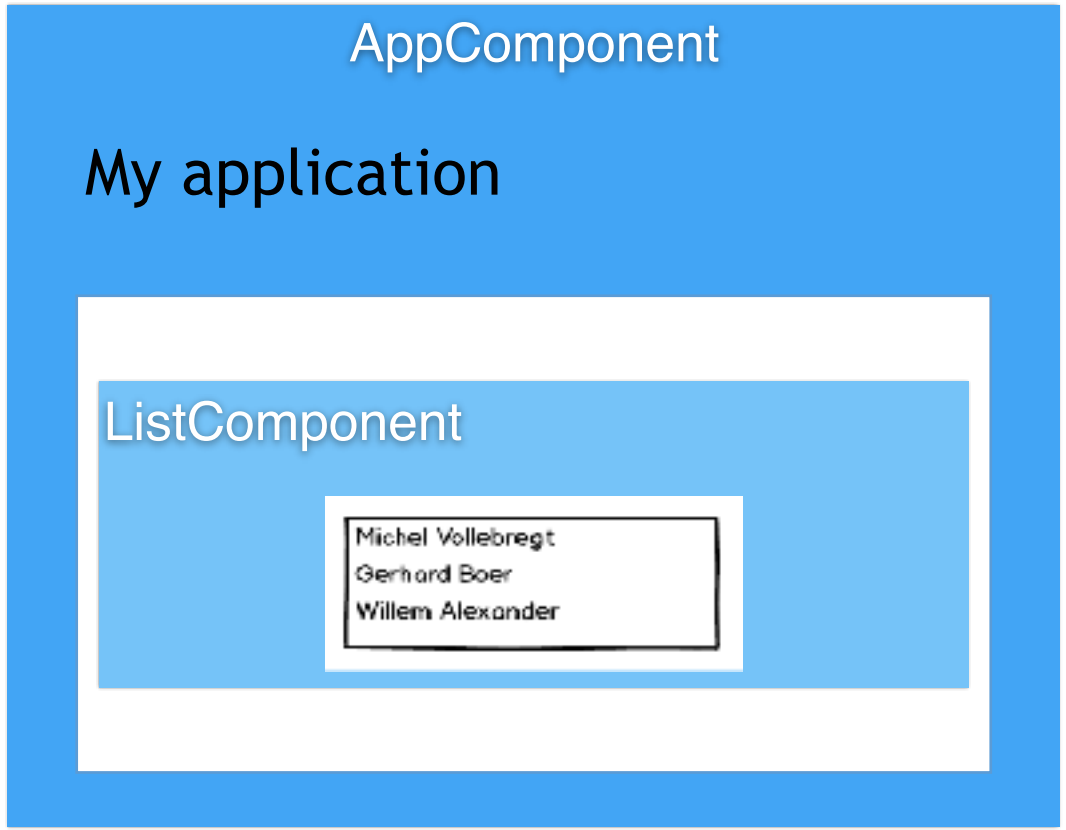
Router outlet




```
<div id="app-component">
  <h1>My application</h1>
  <router-outlet></router-outlet>
</div>
```

Router outlet

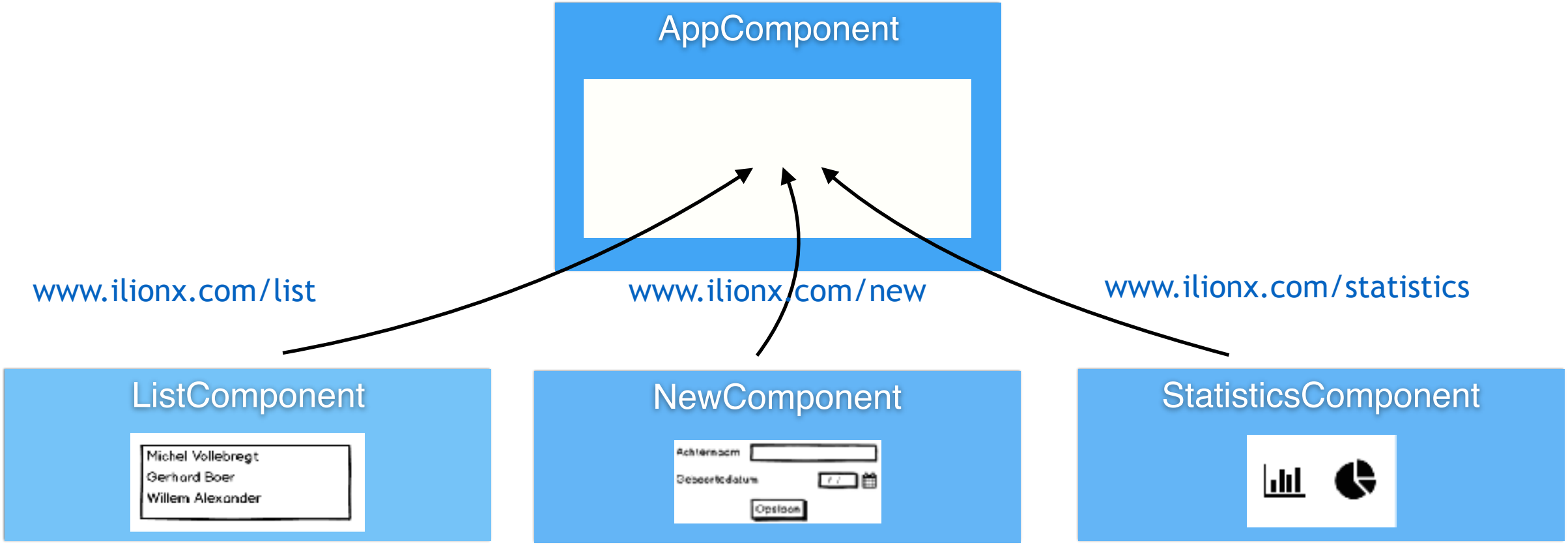
www.ilionx.com/list




```
<div id="app-component">
  <h1>My application</h1>
  <router-outlet></router-outlet>
</div>
```

ilionx
Your partner in digital business

Router outlet



Router outlet

- Placeholder for component
- Content dependant on URL
- (activate) / (deactivate)

Configuring routes

Define array of Routes

app-routing.module.ts

```
const APP_ROUTES : Route[] = [




];
```

Configuring routes

Define array of Routes

app-routing.module.ts

```
const APP_ROUTES : Route[] = [


    { path: 'list',       component: ListComponent },
    { path: 'new',        component: NewComponent },
    { path: 'statistics', component: StatisticsComponent }

];
```

Configuring routes

Redirect from empty path, with pathMatch

app-routing.module.ts

```
const APP_ROUTES : Route[] = [

    { path: '',           redirectTo: 'list', pathMatch: 'full' },

    { path: 'list',       component: ListComponent },
    { path: 'new',        component: NewComponent },
    { path: 'statistics', component: StatisticsComponent }

];
```

Configuring routes

Export AppRoutingModule

app-routing.module.ts

```
const APP_ROUTES : Route[] = [

    { path: '',           redirectTo: 'list', pathMatch: 'full' },

    { path: 'list',       component: ListComponent },
    { path: 'new',        component: NewComponent },
    { path: 'statistics', component: StatisticsComponent }

];

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

Initialize routes

Import AppRoutingModule in AppModule

app.module.ts

```
@NgModule({
   declarations: [ ContactsComponent ],
   imports: [],
   exports: []
})
export class AppModule {


}
```

Initialize routes

Importeer AppRoutingModule in AppModule

app.module.ts

```
@NgModule({
   declarations: [ ContactsComponent ],
   imports: [ AppRoutingModule ],
   exports: []
})
export class AppModule {


}
```

Recap: Routing configuration

- `<router-outlet>`
- `app-routing.module.ts`
- `RouterModule.forRoot()`

Route parameters

…/contact/12

```
{ path: 'contact/:id',        component: NewComponent }
```

Getting Route parameters

3 examples

route.component.ts

```
constructor(private activatedRoute: ActivatedRoute) { }

ngOnInit() {
  let id:string = this.activatedRoute.snapshot.params['id'];
  this.handleRoute(id);

  this.activatedRoute.params
    .subscribe((params: Params) => this.handleRoute(params['id']));

  this.activatedRoute.params
    .pluck('id')
    .subscribe((id: string) => this.handleRoute(id);
}
```

Getting route parameters

Inject ActivatedRoute

route.component.ts

```
constructor(private activatedRoute: ActivatedRoute) { }

ngOnInit() {
  let id:string = this.activatedRoute.snapshot.params['id'];
  this.handleRoute(id);

  this.activatedRoute.params
    .subscribe((params: Params) => this.handleRoute(params['id']));

  this.activatedRoute.params
    .pluck('id')
    .subscribe((id: string) => this.handleRoute(id);
}
```

Getting route parameters

Snapshot gives you the value at that time

route.component.ts

```
constructor(private activatedRoute: ActivatedRoute) { }

ngOnInit() {
  let id:string = this.activatedRoute.snapshot.params['id'];
  this.handleRoute(id);

  this.activatedRoute.params
    .subscribe((params: Params) => this.handleRoute(params['id']));

  this.activatedRoute.params
    .pluck('id')
    .subscribe((id: string) => this.handleRoute(id);
}
```

Getting route parameters

Subscribe to reuse the component

route.component.ts

```
constructor(private activatedRoute: ActivatedRoute) { }

ngOnInit() {
  let id:string = this.activatedRoute.snapshot.params['id'];
  this.handleRoute(id);

  this.activatedRoute.params
    .subscribe((params: Params) => this.handleRoute(params['id']));

  this.activatedRoute.params
    .pluck('id')
    .subscribe((id: string) => this.handleRoute(id);
}
```

Getting route parameters

pluck operator

route.component.ts

```
constructor(private activatedRoute: ActivatedRoute) { }

ngOnInit() {
  let id:string = this.activatedRoute.snapshot.params['id'];
  this.handleRoute(id);

  this.activatedRoute.params
    .subscribe((params: Params) => this.handleRoute(params['id']));

  this.activatedRoute.params.pipe(
      pluck('id')
    ).subscribe((id: string) => this.handleRoute(id);
}
```

Router service

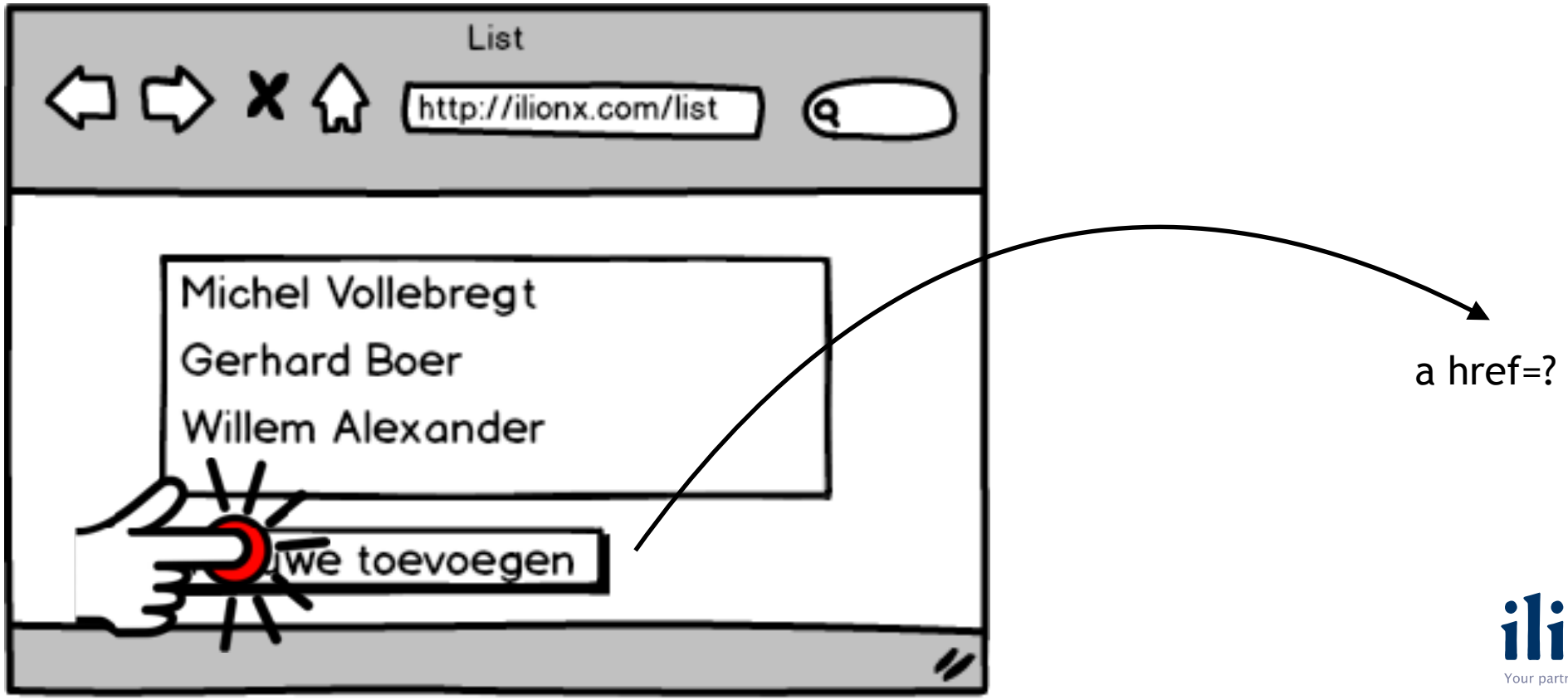
Navigate imperatively

```
router.navigate([‘/contact’, 12])

router.navigate([…], { queryParams: {…} })

router.routerState.queryParams.subscribe(…)
```

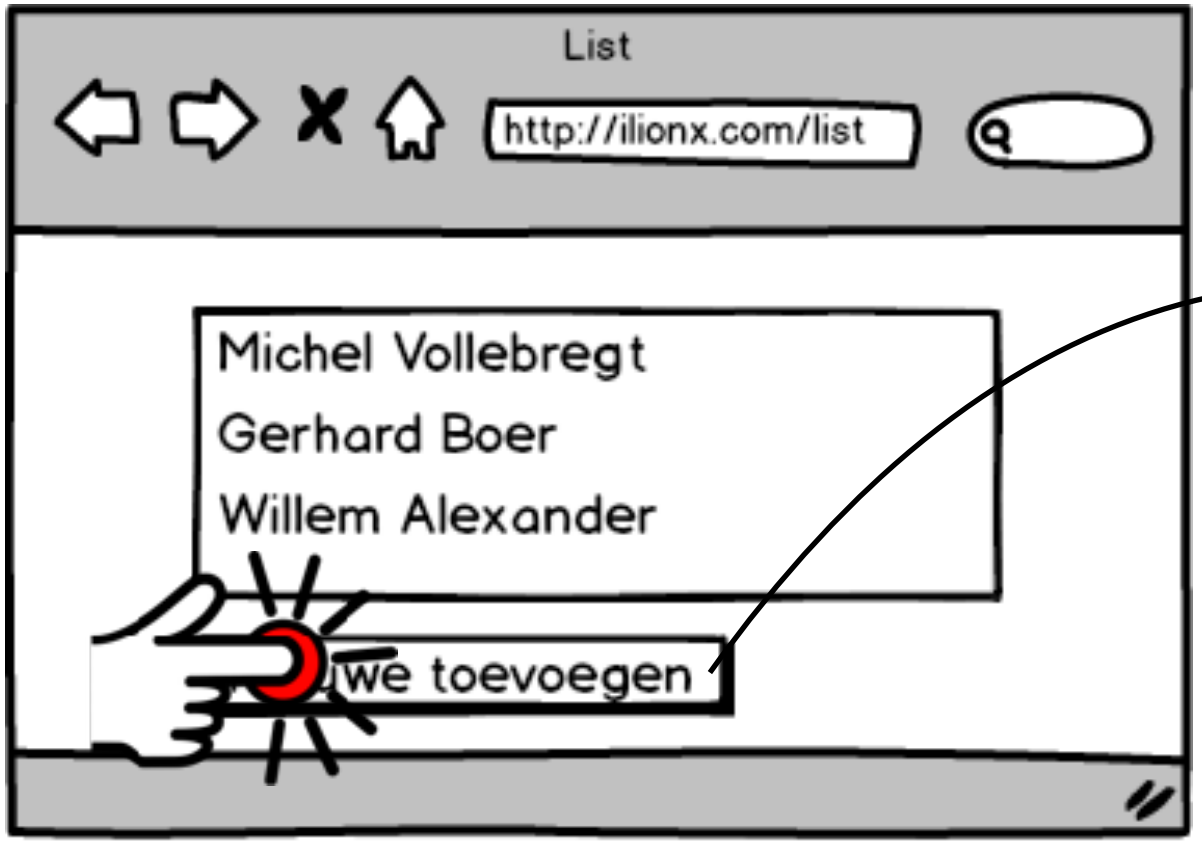
Recap: Router parameters

- :param
- ActivatedRoute
- Router

Router link


List
http://ilionx.com/list
Michel Vollebregt
Gerhard Boer
Willem Alexander
we toevoegen
a href=?

Router link



router link

```
<div ...>

  <a [routerLink]="['/new']">
    Add
  </a>

</div>
```

ilionx
Your partner in digital business

Router link

RouterLink points to the path in Routes[]

```
{ path: 'list',       component: ListComponent },
{ path: 'new',        component: NewComponent },
{ path: 'statistics', component: StatisticsComponent }
```

```
<div ...>

  <a [routerLink]="['/new']">
    Add
  </a>

</div>
```

Show the active path

```
<a routerLink="/new" routerLinkActive="cssClass">

<a routerLink="/new" [routerLinkActive]="['css1', 'css2']">
```

directive controls the class based on URL

Look back router link

- href attribute
- routerLink
- routerLinkActive

Child routes

Combining routes

```
const APP_ROUTES : Route[] = [
    { path: '',           redirectTo: 'list', pathMatch: 'full' },
    { path: 'list',       component: ListComponent },
    { path: 'new',        component: NewComponent },

    { path: 'statistics', component: StatisticsComponent }




];
```

Child routes

Combining routes

```
const APP_ROUTES : Route[] = [
    { path: '',           redirectTo: 'list', pathMatch: 'full' },
    { path: 'list',       component: ListComponent },
    { path: 'new',        component: NewComponent },

    { path: 'statistics', component: StatisticsComponent }

    { path: ‘statistics/bar', component: BarComponent }
    { path: ‘statistics/pie‘, component: PieComponent }
    { path: ‘statistics/line', component: LineComponent }
    { path: ‘statistics/column', component: ColumnComponent }
    { path: ‘statistics/scatter', component: ScatterComponent }
];
```

Child routes

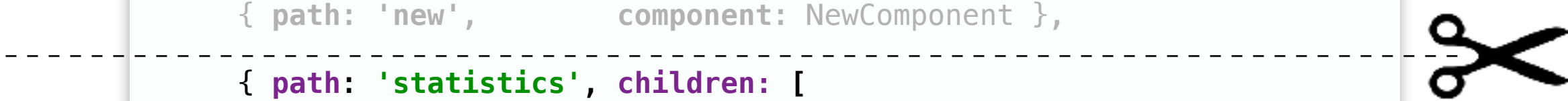
Combining routes

```
const APP_ROUTES : Route[] = [
    { path: '',           redirectTo: 'list', pathMatch: 'full' },
    { path: 'list',       component: ListComponent },
    { path: 'new',        component: NewComponent },

    { path: 'statistics', children: [
        { path: '', component: StatisticsComponent }
        { path: 'bar', component: BarComponent }
        { path: 'pie', component: PieComponent }
        { path: 'line', component: LineComponent }
        { path: 'column', component: ColumnComponent }
        { path: 'scatter', component: ScatterComponent }
]];
```

Routes per module

Split the route definitions

```
const APP_ROUTES : Route[] = [
    { path: '',           redirectTo: 'list', pathMatch: 'full' },
    { path: 'list',       component: ListComponent },
    { path: 'new',        component: NewComponent },
    { path: 'statistics', children: [
        { path: '', component: StatisticsComponent }
        { path: 'bar', component: BarComponent }
        { path: 'pie', component: PieComponent }
        { path: 'line', component: LineComponent }
        { path: 'column', component: ColumnComponent }
        { path: 'scatter', component: ScatterComponent }
]];
```



Routes per module

Define statistics routes with the statics module

```
const APP_ROUTES : Route[] = [
    { path: '',           redirectTo: 'list', pathMatch: 'full' },
    { path: 'list',       component: ListComponent },
    { path: 'new',        component: NewComponent }];
```

```
const STATISTICS_ROUTES : Route[] = [
    { path: 'statistics', children: [
        { path: '', component: StatisticsComponent }
        { path: 'bar', component: BarComponent }
        { path: 'pie', component: PieComponent }
        { path: 'line', component: LineComponent }
        { path: 'column', component: ColumnComponent }
        { path: 'scatter', component: ScatterComponent }
]];
```

Routes per module

Use RouterModule.forChild

statistics-routing.module.ts

```
const STATISTICS_ROUTES : Route[] = [
    { path: 'statistics', children: [
        { path: '', component: StatisticsComponent }
        { path: 'bar', component: BarComponent }
        { path: 'pie', component: PieComponent }
        { path: 'line', component: LineComponent }
        { path: 'column', component: ColumnComponent }
        { path: 'scatter', component: ScatterComponent }
]];
```

Routes per module

Use RouterModule.forChild

statistics-routing.module.ts

```
const STATISTICS_ROUTES : Route[] = [
    { path: 'statistics', children: [
        { path: '', component: StatisticsComponent }
        { path: 'bar', component: BarComponent }
        { path: 'pie', component: PieComponent }
        { path: 'line', component: LineComponent }
        { path: 'column', component: ColumnComponent }
        { path: 'scatter', component: ScatterComponent }
]];

export const StatisticsRoutingModule =
                        RouterModule.forChild(STATISTICS_ROUTES);
```

Lazy loading

Using loadChildren

```
const APP_ROUTES : Route[] = [
    { path: '',           redirectTo: 'list', pathMatch: 'full' },
    { path: 'list',       component: ListComponent }




];
```

```
const STATISTICS_ROUTES : Route[] = [
    { path: 'statistics', children: [
        { path: ‘’, component: StatisticsComponent }
        { path: ‘bar', component: BarComponent }
        { path: ‘pie‘, component: PieComponent }
]];
```

Lazy loading

Using loadChildren

```
const APP_ROUTES : Route[] = [
    { path: '',           redirectTo: 'list', pathMatch: 'full' },
    { path: 'list',       component: ListComponent },

    { path: 'statistics',
      loadChildren: 'app/statistics/statistics.module#StatisticsModule'
    }
```

```
const STATISTICS_ROUTES : Route[] = [

        { path: '', component: StatisticsComponent }
        { path: 'bar', component: BarComponent }
        { path: 'pie', component: PieComponent }
];
```

Recap: Routing

- Mapping URL - component
- Routing configurable per module
- Easy lazy loading
- router-outlet, routerLink, routerLinkActive

And more

- Guards
  - CanActivate
  - CanDeactivate
- Prefetching

Routing - Demo

ilionx
Your partner in digital business