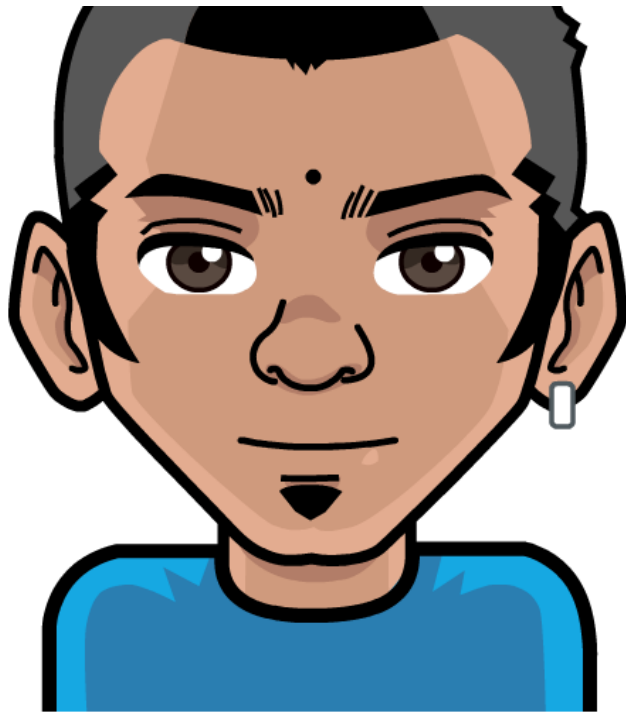


RAJU GANDHI

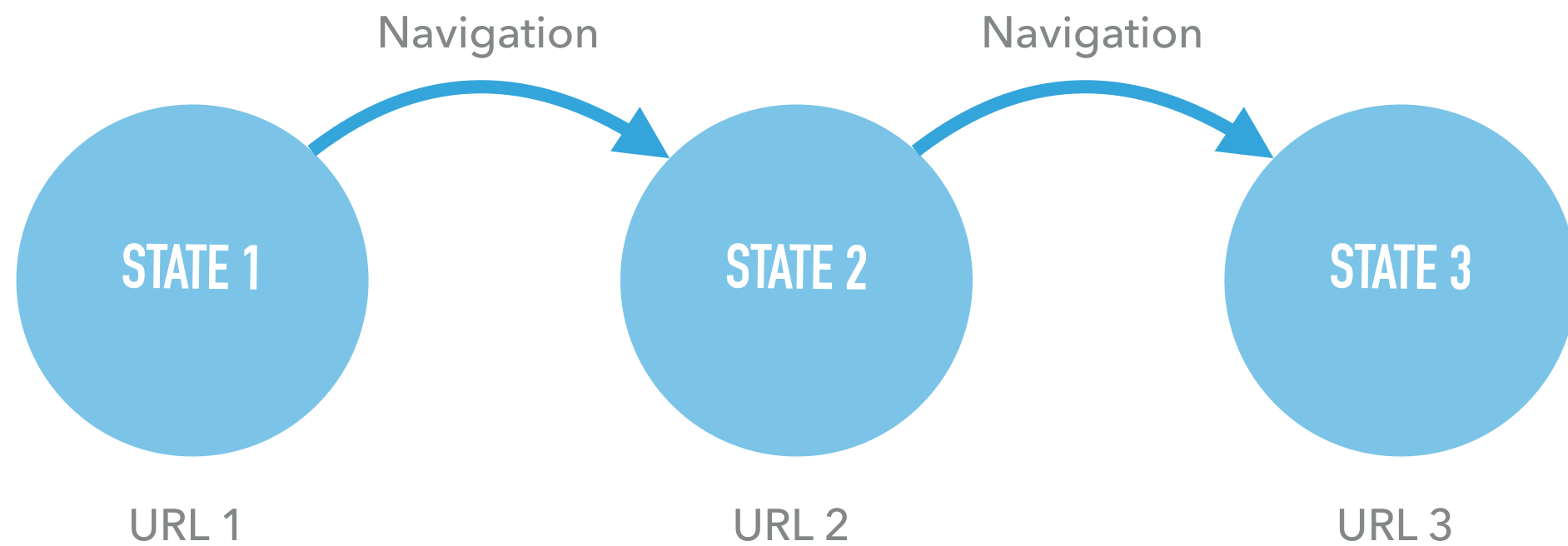
NAVIGATING THE ANGULAR ROUTER



RAJU GANDHI

   @LOOSELYTYPED
CTO - INTEGRALLIS SOFTWARE

VISUALIZING ROUTER STATE



**AN ARRANGEMENT OF
COMPONENTS THAT COMPOSE
THE CURRENT VIEW**

@looselytyped

CONFIGURATION

```
export const routes: Routes = [
  {
    path: 'home',
    component: HomeComponent,
    data: { title: "Welcome!" }
  },
  {
    path: 'classes',
    component: ClassesComponent,
    data: { title: "Classes" },
    resolve: { classes: ClassesResolver },
    children: [
      {
        path: ':id',
        children: [
          {
            path: '',
            component: ClassComponent,
            resolve: { class: 'classResolver' }
          },
          {
            path: '',
            component: DescriptionsComponent,
            resolve: { descriptions: 'descriptionsResolver' },
            outlet: 'descriptions'
          }
        ]
      }
    ]
  },
  {
    path: '**', redirectTo: '/home', pathMatch: 'full'
  }
]
```

```
export interface Route {  
  path?: string;  
  pathMatch?: string;  
  matcher?: UrlMatcher;  
  component?: Type<any>;  
  redirectTo?: string;  
  outlet?: string;  
  canActivate?: any[];  
  canActivateChild?: any[];  
  canDeactivate?: any[];  
  canLoad?: any[];  
  data?: Data;  
  resolve?: ResolveData;  
  children?: Routes;  
  loadChildren?: LoadChildren;  
}
```

HOW TO MATCH

```
export interface Route {  
  path?: string;  
  pathMatch?: string;  
  matcher?: UrlMatcher;  
  component?: Type<any>;  
  redirectTo?: string;  
  outlet?: string;  
  canActivate?: any[];  
  canActivateChild?: any[];  
  canDeactivate?: any[];  
  canLoad?: any[];  
  data?: Data;  
  resolve?: ResolveData;  
  children?: Routes;  
  loadChildren?: LoadChildren;  
}
```



```
export interface Route {  
  path?: string;  
  pathMatch?: string;  
  matcher?: UrlMatcher;  
  component?: Type<any>;  
  redirectTo?: string;  
  outlet?: string;  
  canActivate?: any[];  
  canActivateChild?: any[];  
  canDeactivate?: any[];  
  canLoad?: any[];  
  data?: Data;  
  resolve?: ResolveData;  
  children?: Routes;  
  loadChildren?: LoadChildren;  
}
```



WHAT TO
DO

```
export interface Route {  
  path?: string;  
  pathMatch?: string;  
  matcher?: UrlMatcher;  
  component?: Type<any>;  
  redirectTo?: string;  
  outlet?: string;  
  canActivate?: any[];  
  canActivateChild?: any[];  
  canDeactivate?: any[];  
  canLoad?: any[];  
  data?: Data;  
  resolve?: ResolveData;  
  children?: Routes;  
  loadChildren?: LoadChildren;  
}
```



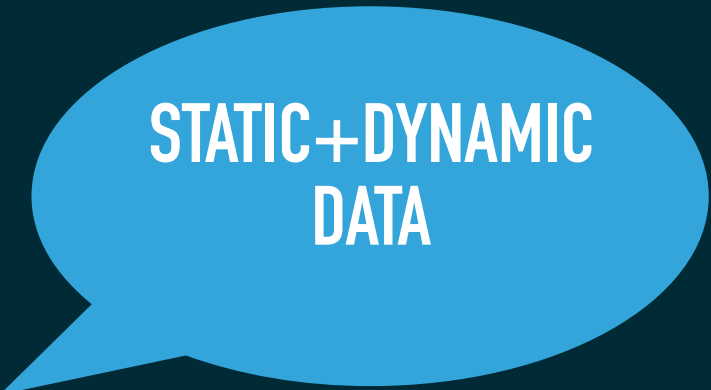
WHERE IN THE
DOM?

```
export interface Route {  
  path?: string;  
  pathMatch?: string;  
  matcher?: UrlMatcher;  
  component?: Type<any>;  
  redirectTo?: string;  
  outlet?: string;  
  canActivate?: any[];  
  canActivateChild?: any[];  
  canDeactivate?: any[];  
  canLoad?: any[];  
  data?: Data;  
  resolve?: ResolveData;  
  children?: Routes;  
  loadChildren?: LoadChildren;  
}
```



GUARDS

```
export interface Route {  
  path?: string;  
  pathMatch?: string;  
  matcher?: UrlMatcher;  
  component?: Type<any>;  
  redirectTo?: string;  
  outlet?: string;  
  canActivate?: any[];  
  canActivateChild?: any[];  
  canDeactivate?: any[];  
  canLoad?: any[];  
  data?: Data;  
  resolve?: ResolveData;  
  children?: Routes;  
  loadChildren?: LoadChildren;  
}
```



STATIC+DYNAMIC
DATA

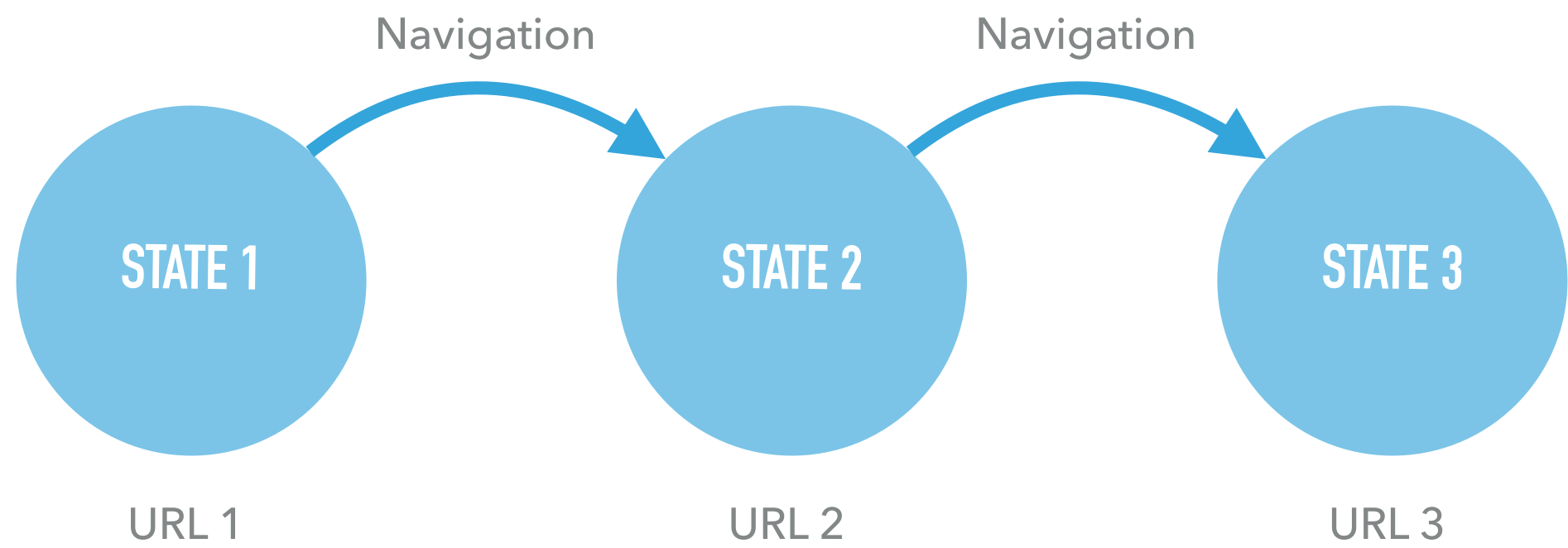
```
export interface Route {  
  path?: string;  
  pathMatch?: string;  
  matcher?: UrlMatcher;  
  component?: Type<any>;  
  redirectTo?: string;  
  outlet?: string;  
  canActivate?: any[];  
  canActivateChild?: any[];  
  canDeactivate?: any[];  
  canLoad?: any[];  
  data?: Data;  
  resolve?: ResolveData;  
  children?: Routes;  
  loadChildren?: LoadChildren;  
}
```

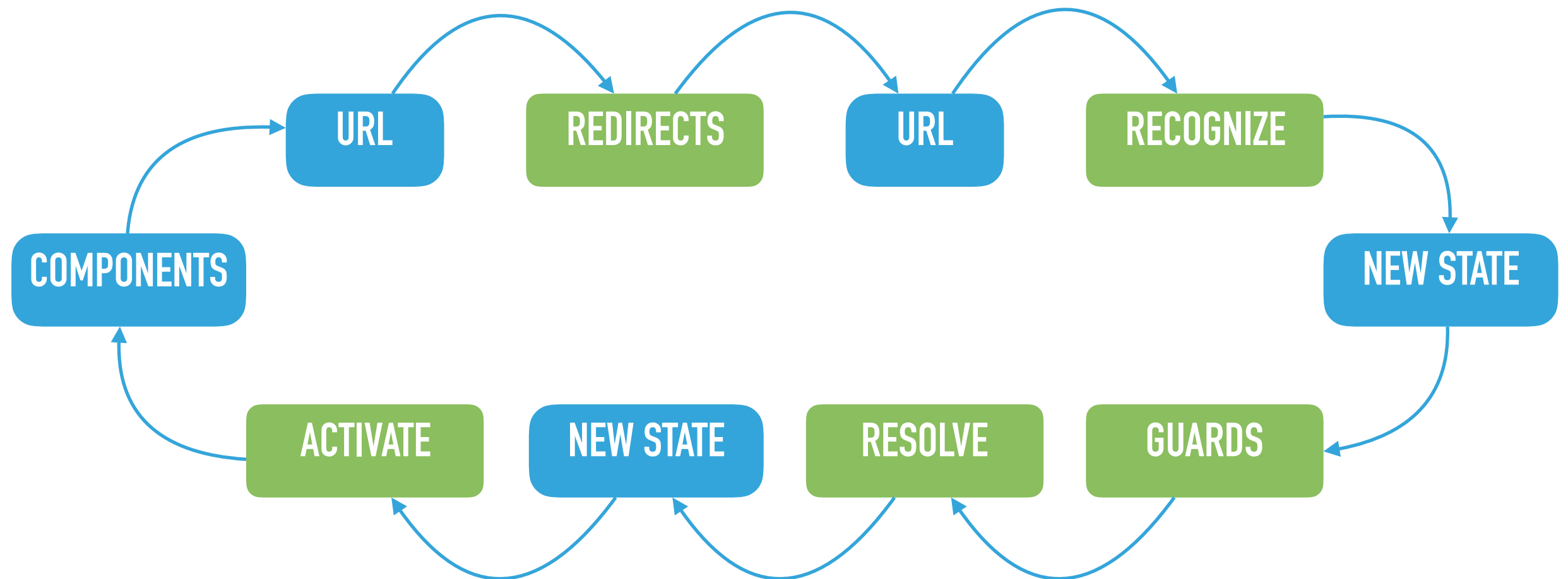


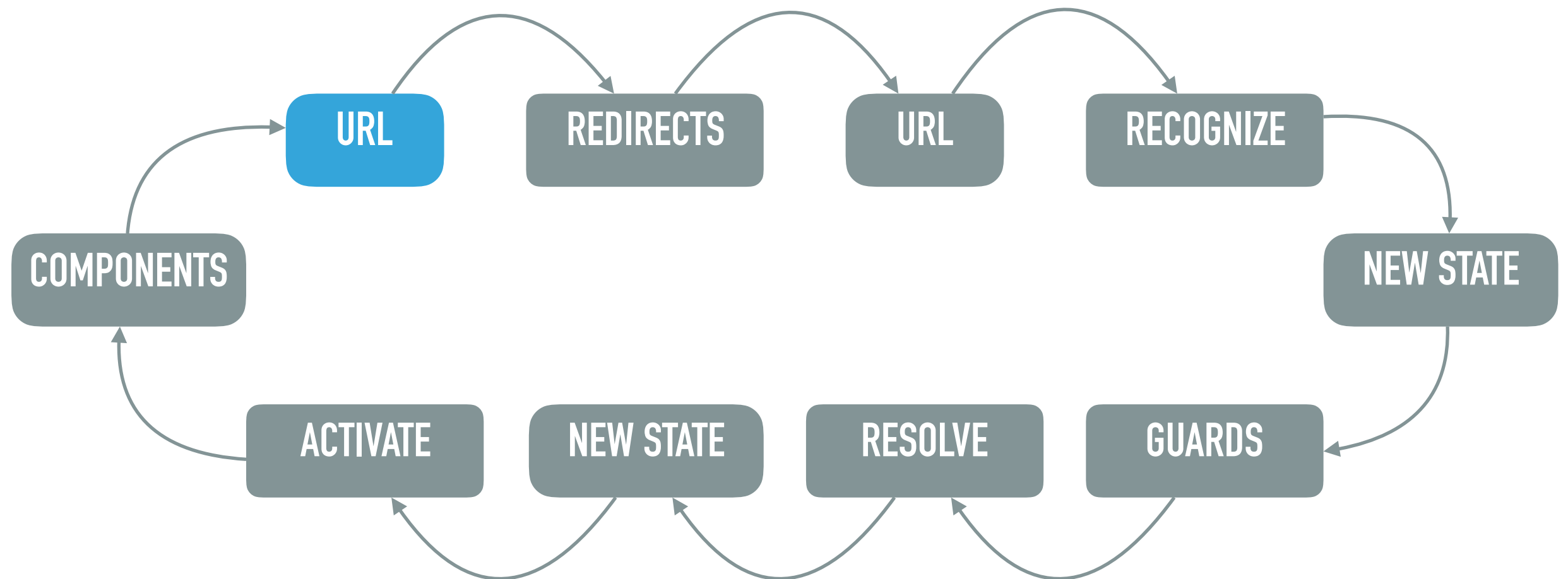
CHILD ROUTES

WORKFLOW

VISUALIZING ROUTER STATE

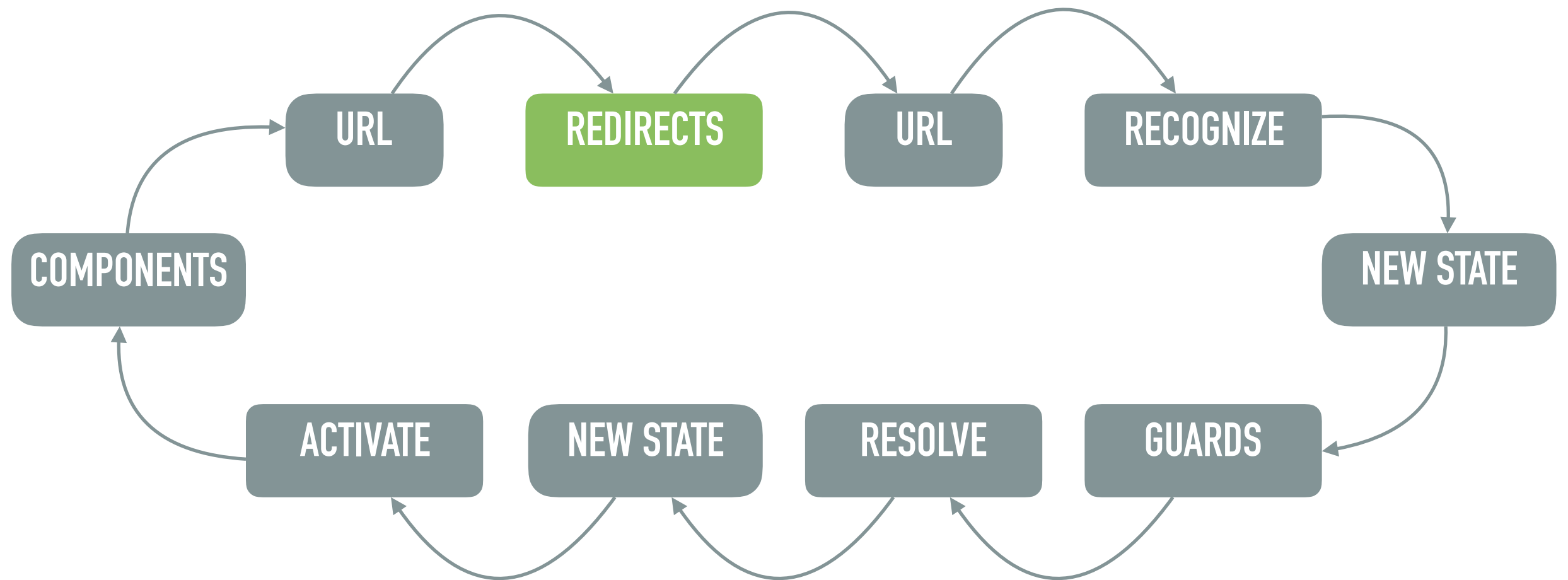






TYPES

- ▶ Path
- ▶ Parameterized
- ▶ Query Params
- ▶ Matrix Params
 - ▶ `/path/...;a=1;a=2/childPath;b=1;b=2`
- ▶ Secondary Segments (Auxiliary)
 - ▶ `/path(aside:detail//sidebar:true)/childPath(popup:compose)`

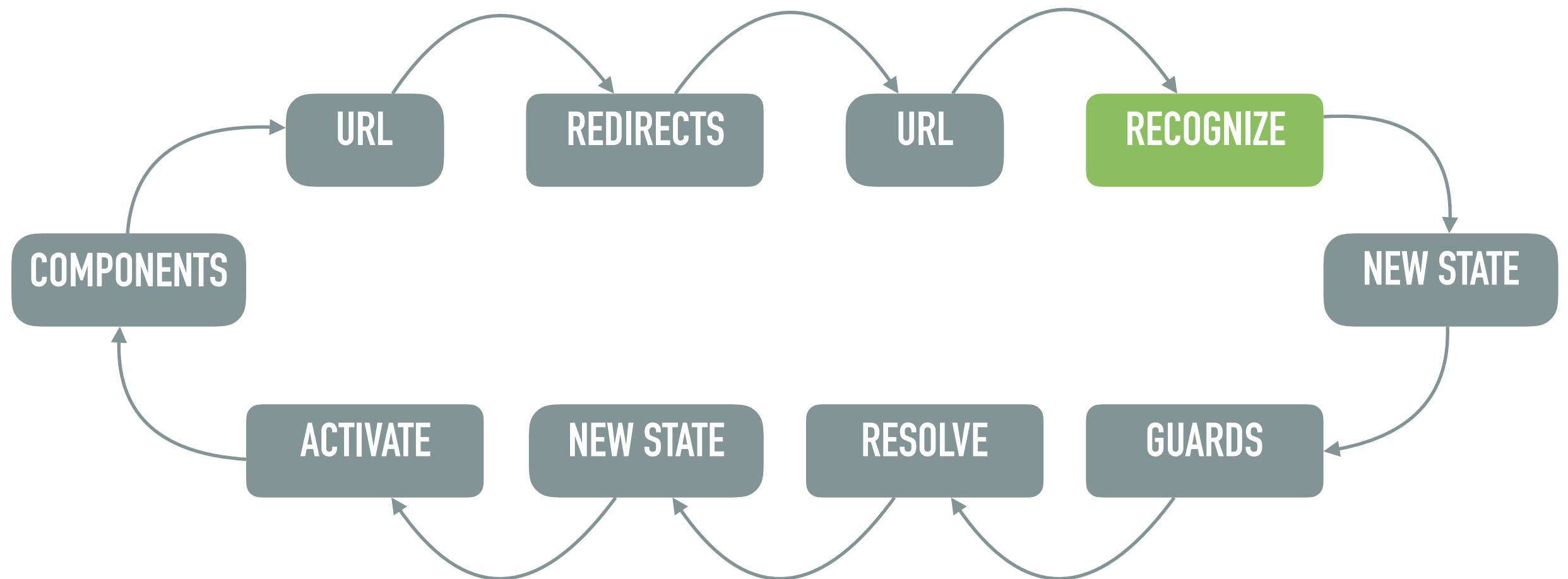


```
export const routes: Routes = [  
  {  
    path: 'home',  
    component: HomeComponent,  
    data: { title: "Welcome!" }  
  },  
  // ...  
  {  
    path: '**', redirectTo: '/home', pathMatch: 'full'  
  }  
]
```

```
export const routes: Routes = [  
  {  
    path: 'home',  
    component: HomeComponent,  
    data: { title: "Welcome!" }  
  },  
  // ...  
  {  
    path: '**', redirectTo: '/home', pathMatch: 'full'  
  }  
]
```



**ORDER
MATTERS!!!**



```
{  
  path: 'home',  
  component: HomeComponent,  
  data: { title: "Welcome!" }  
},  
// ...  
{  
  path: '**', redirectTo: '/home', pathMatch: 'full'  
}
```

/home



MATCH /HOME

```
{  
  path: 'home',  
  component: HomeComponent,  
  data: { title: "Welcome!" }  
},  
// ...  
{  
  path: '**', redirectTo: '/home', pathMatch: 'full'  
}
```


/home/1

MATCH /HOME!!

```
{  
  path: 'home',  
  component: HomeComponent,  
  data: { title: "Welcome!" }  
},  
// ...  
{  
  path: '**', redirectTo: '/home', pathMatch: 'full'  
}
```

/home/1

```
{
  path: 'home',
  component: HomeComponent,
  data: { title: "Welcome!" }
},
// ...
{
  path: '**', redirectTo: '/home', pathMatch: 'full'
}
```

MATCH /1?? 🤪

/home/1

```
{  
  path: 'home',  
  component: HomeComponent,  
  data: { title: "Welcome!" }  
},  
// ...  
{  
  path: '**', redirectTo: '/home', pathMatch: 'full'  
}
```



REDIRECT!

/classes/1

```
{
  path: 'classes',
  component: ClassesComponent,
  data: { title: "Classes" },
  resolve: { classes: ClassesResolver },
  children: [
    {
      path: ':id',
      children: [
        {
          path: '',
          component: ClassComponent,
          resolve: { class: 'classResolver' }
        },
        {
          path: '',
          component: DescriptionsComponent,
          resolve: { descriptions: 'descriptionsResolver' },
          outlet: 'descriptions'
        }
      ]
    }
  ]
}
```

/classes/1

```
{
  path: 'classes',
  component: ClassesComponent,
  data: { title: "Classes" },
  resolve: { classes: ClassesResolver },
  children: [
    {
      path: ':id',
      children: [
        {
          path: '',
          component: ClassComponent,
          resolve: { class: 'classResolver' }
        },
        {
          path: '',
          component: DescriptionsComponent,
          resolve: { descriptions: 'descriptionsResolver' },
          outlet: 'descriptions'
        }
      ]
    }
  ]
}
```

MATCH /CLASSES!!

/classes/1

```
{
  path: 'classes',
  component: ClassesComponent,
  data: { title: 'Classes' },
  resolve: { classes: Classes },
  children: [
    {
      path: ':id',
      children: [
        {
          path: '',
          component: ClassComponent,
          resolve: { class: 'classResolver' }
        },
        {
          path: '',
          component: DescriptionsComponent,
          resolve: { descriptions: 'descriptionsResolver' },
          outlet: 'descriptions'
        }
      ]
    }
  ]
}
```

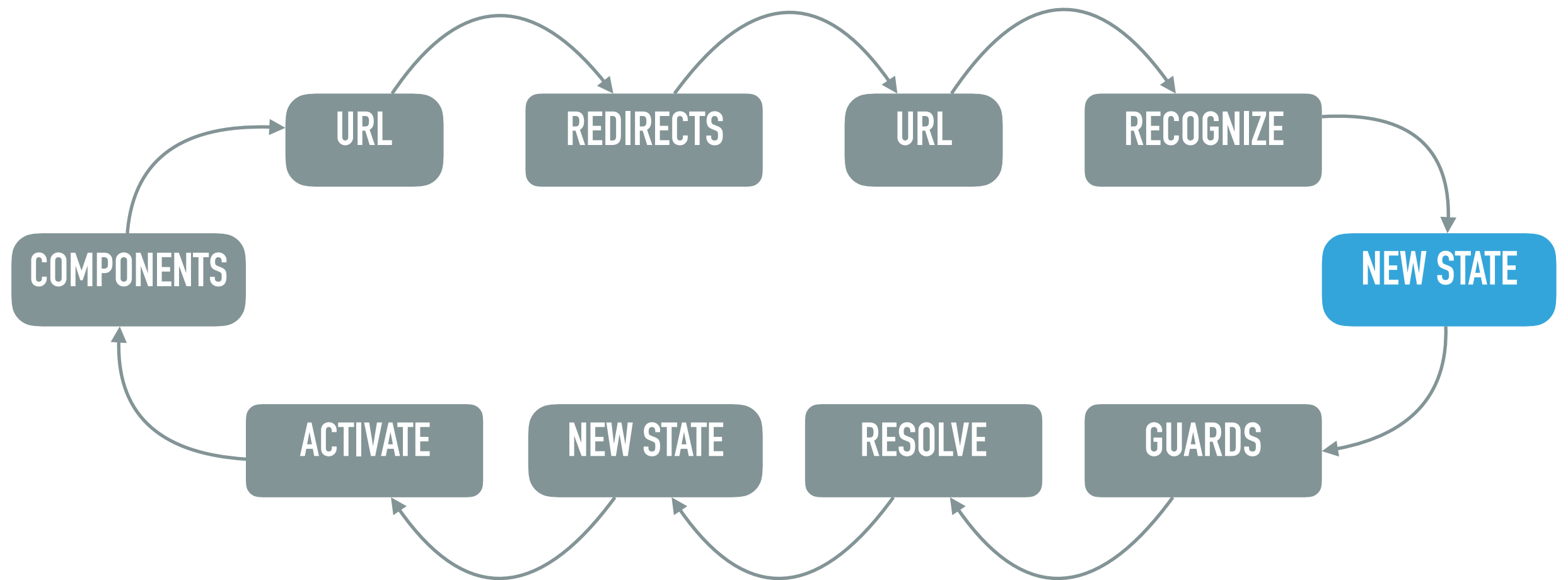
MATCH /1!!

```
/classes/1
```

```
{  
  path: 'classes',  
  component: ClassesComponent,  
  data: { title: "Classes" },  
  resolve: { classes: ClassesResolver },  
  children: [  
    {  
      path: ':id',  
      children: [  
        {  
          path: '',  
          component: ClassComponent,  
          resolve: { class: 'classResolver' }  
        },  
        {  
          path: '',  
          component: DescriptionsComponent,  
          resolve: { descriptions: 'descriptionsResolver' },  
          outlet: 'descriptions'  
        }  
      ]  
    }  
  ]  
}
```

MATCH /1!!

**ACTIVATES 2
CHILDREN**



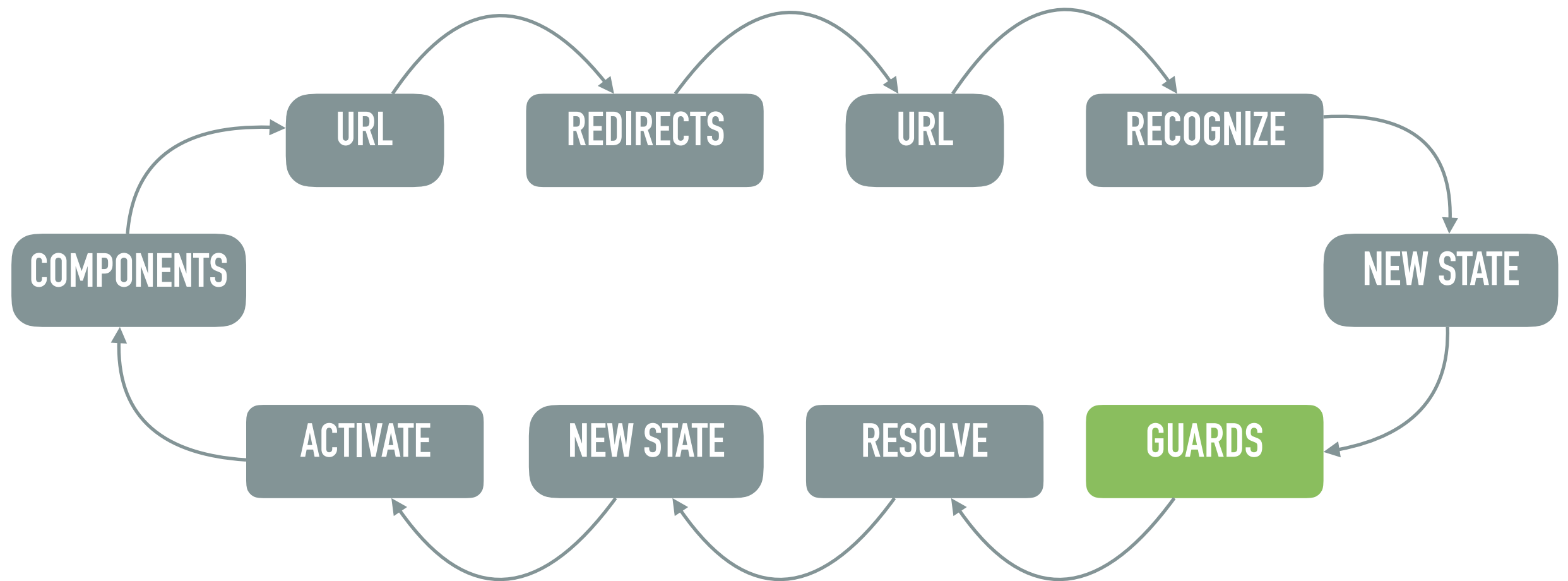

```
component: root  
url: ''  
data: {}  
params: {}
```

```
graph TD; A["component: root<br/>url: ''<br/>data: {}<br/>params: {}"] --> B["component: ClassesComponent<br/>url: 'classes'<br/>data: { title: '...' }<br/>params: {}<br/>outlet: 'primary'"]; B --> C["component: ClassComponent<br/>url: ''<br/>data: {}<br/>params: { id: '1' }<br/>outlet: 'primary'"]; B --> D["component: DescriptionsComponent<br/>url: ''<br/>data: {}<br/>params: { id: '1' }<br/>outlet: 'descriptions'"];
```

```
component: ClassesComponent  
url: 'classes'  
data: { title: '...' }  
params: {}  
outlet: 'primary'
```

```
component: ClassComponent  
url: ''  
data: {}  
params: { id: '1' }  
outlet: 'primary'
```

```
component: DescriptionsComponent  
url: ''  
data: {}  
params: { id: '1' }  
outlet: 'descriptions'
```

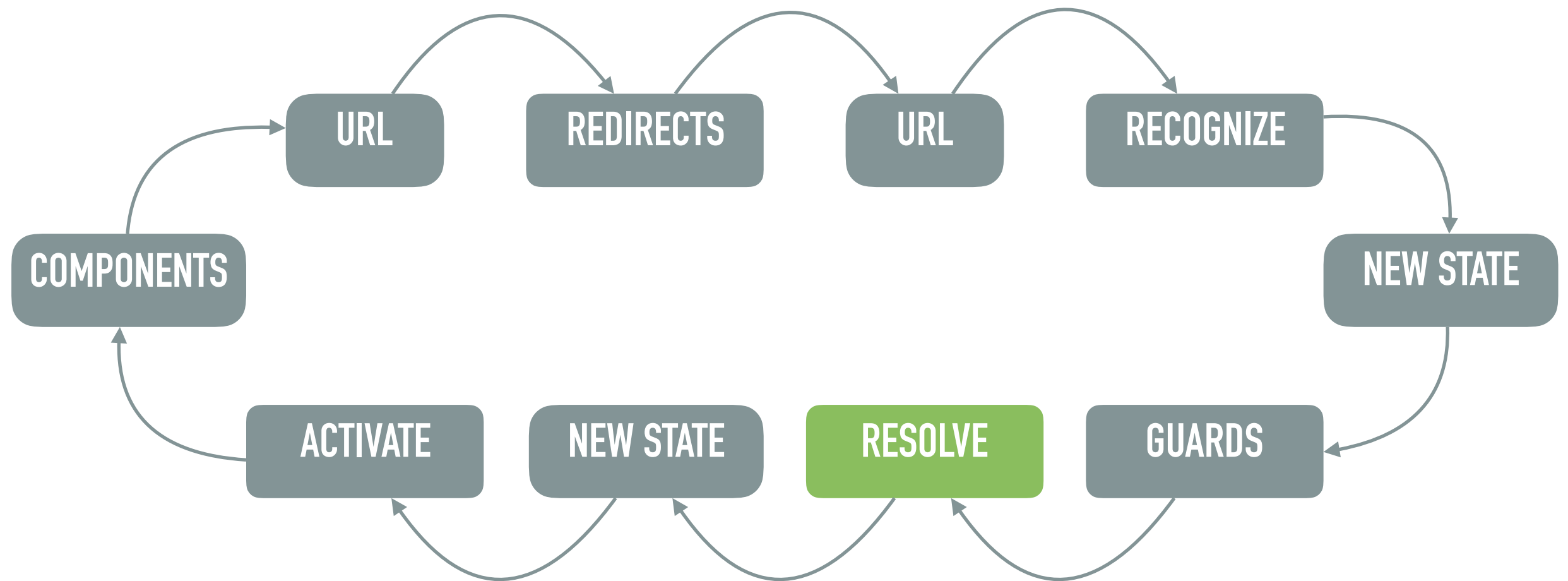


SPECIFICS

- ▶ Can be a Provider, or a simple function
- ▶ Predicate that returns `Observable<boolean>`, `Promise<boolean>` or a simple `boolean`
- ▶ You can supply an `Array` of guards for each state

TYPES

- ▶ CanLoad
 - ▶ Specifically for lazy-loaded modules
- ▶ CanActivate
 - ▶ Check to see if a route can be activated
- ▶ CanActivateChild
 - ▶ Check to see if the child routes of a route can be activated
- ▶ CanDeactivate
 - ▶ Decide to see if a route can be deactivated



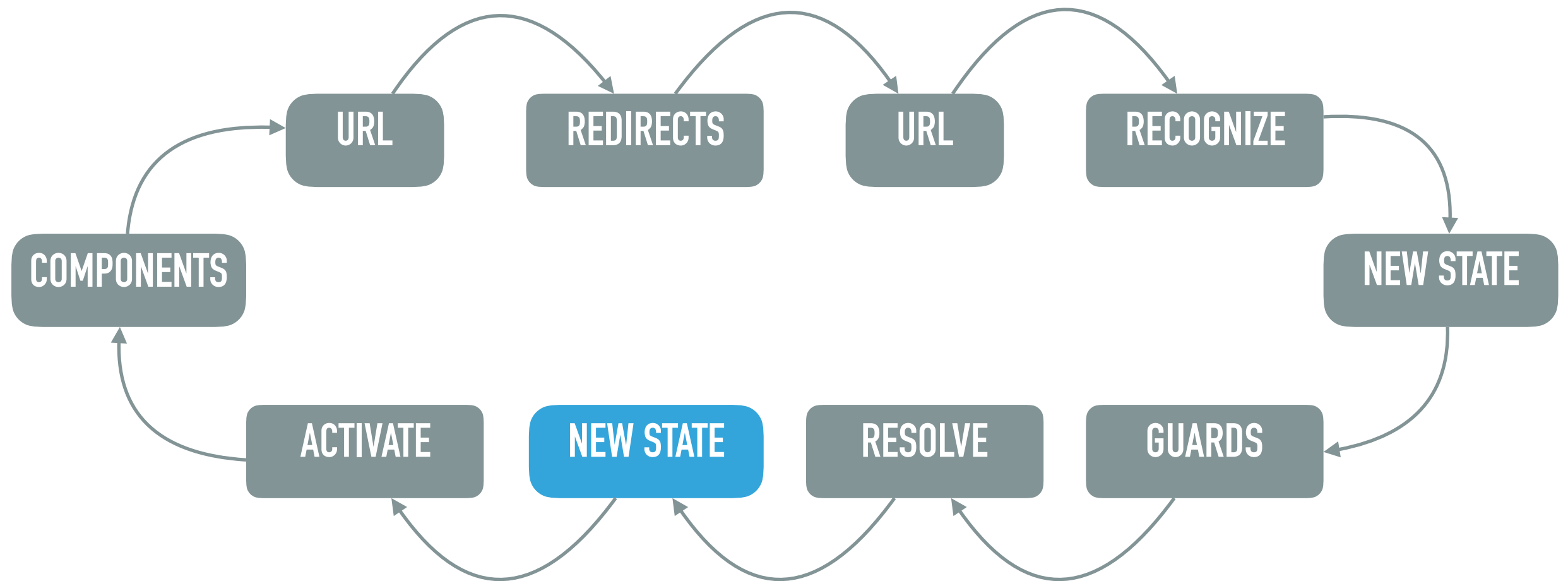
```
{
  path: 'classes',
  component: ClassesComponent,
  data: {
    title: "Classes"
  },
  resolve: {
    classes: ClassesResolver
  }
}
```

```
@Injectable()
export class ClassesResolver implements Resolve<any> {
  resolve(route: ActivatedRouteSnapshot,
    state: RouterStateSnapshot): Observable<any> {
    return Observable.of({
      msg: "Data!!"
    });
  }
}
```

```
export function classResolver(repo: RepoService) {  
  return (route: ActivatedRouteSnapshot) => {  
    return repo.getClass(+route.params['id'])  
  }  
}
```

```
{  
  path: '',  
  component: ClassComponent,  
  resolve: {  
    class: 'classResolver'  
  }  
}
```

```
@NgModule({  
  declarations: [  
  ],  
  imports: [  
  ],  
  providers: [  
    { provide: 'classResolver', useFactory: classResolver,  
      deps: [RepoService] }  
  ],  
  bootstrap: [AppComponent]  
})
```




```
component: root  
url: ''  
data: {}  
params: {}
```

```
graph TD; A["component: root<br/>url: ''<br/>data: {}<br/>params: {}"] --> B["component: ClassesComponent<br/>url: 'classes'<br/>data: { title: '...' }<br/>params: {}<br/>outlet: 'primary'"]; B --> C["component: ClassComponent<br/>url: ''<br/>data: {}<br/>params: { id: '1' }<br/>outlet: 'primary'"]; B --> D["component: DescriptionsComponent<br/>url: ''<br/>data: {}<br/>params: { id: '1' }<br/>outlet: 'descriptions'"];
```

```
component: ClassesComponent  
url: 'classes'  
data: { title: '...' }  
params: {}  
outlet: 'primary'
```

```
component: ClassComponent  
url: ''  
data: {}  
params: { id: '1' }  
outlet: 'primary'
```

```
component: DescriptionsComponent  
url: ''  
data: {}  
params: { id: '1' }  
outlet: 'descriptions'
```

```
component: root  
url: ''  
data: {}  
params: {}
```

```
graph TD; A["component: root<br/>url: ''<br/>data: {}<br/>params: {}"] --> B["component: ClassesComponent<br/>url: 'classes'<br/>data: { title: '...', classes: ...}<br/>params: {}<br/>outlet: 'primary'"]; B --> C["component: ClassComponent<br/>url: ''<br/>data: { class: ... }<br/>params: { id: '1' }<br/>outlet: 'primary'"]; B --> D["component: DescriptionsComponent<br/>url: ''<br/>data: { descriptions: ... }<br/>params: { id: '1' }<br/>outlet: 'descriptions'"];
```

```
component: ClassesComponent  
url: 'classes'  
data: { title: '...', classes: ...}  
params: {}  
outlet: 'primary'
```

```
component: ClassComponent  
url: ''  
data: { class: ... }  
params: { id: '1' }  
outlet: 'primary'
```

```
component: DescriptionsComponent  
url: ''  
data: { descriptions: ... }  
params: { id: '1' }  
outlet: 'descriptions'
```

EVENTS

TYPES

- ▶ `NavigationStart`
- ▶ `NavigationEnd`
- ▶ `NavigationCancel`
- ▶ `NavigationError`
- ▶ `RoutesRecognized`

```
export class AppComponent {
  loading = false;

  constructor(private router: Router) {
    router.events.subscribe((e: Event) => {
      this.navigationInterceptor(e);
    });
  }

  navigationInterceptor(event: Event): void {
    this.loading = (this.isStart(event) ? true : false);
  }

  isStart(event: Event): boolean {
    return event instanceof NavigationStart;
  }

  isEnd(event: Event): boolean {
    return (event instanceof NavigationEnd)
      || (event instanceof NavigationCancel)
      || (event instanceof NavigationError);
  }
}

<div class="loading-overlay" *ngIf="loading">
  <md-progress-spinner mode="indeterminate"></md-progress-spinner>
</div>
```

DEBUGGING

Router Event: NavigationStart

NavigationStart(id: 1, url: '/home')

Router Event: RoutesRecognized

RoutesRecognized(id: 1, url: '/home', urlAfterRedirects: '/home',
state: Route(url:'', path: '')
{ Route(url:'home', path:'home') })

Router Event: NavigationEnd

NavigationEnd(id: 1, url: '/home', urlAfterRedirects: '/home')

OUTLET

- ▶ Place holder in the DOM to locate a component
- ▶ Can be named

ALL DONE

THANKS!!

RAJU GANDHI

RESOURCES

- ▶ Angular Docs [Router and Navigation](#)
- ▶ Angular Router by [Victor Savkin](#)