



# Angular Performance Tuning

Manfred Steyer  
**SOFTWARE**architekt.at

**Turbo Button**



## Quick Wins

Bundling

Minification

enableProdMode()

**SOFTWARE**architekt.at

## Contents

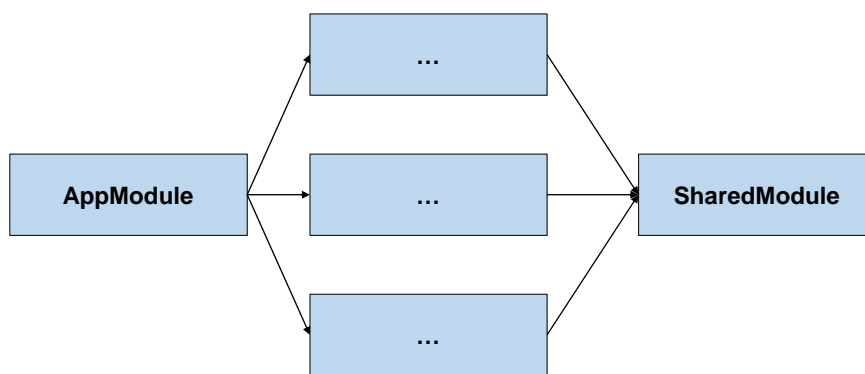
- Lazy Loading and Preloading
- Performance for Data Binding with OnPush
- AOT and Tree Shaking

**SOFTWARE**architekt.at

## Lazy Loading



## Module Structure

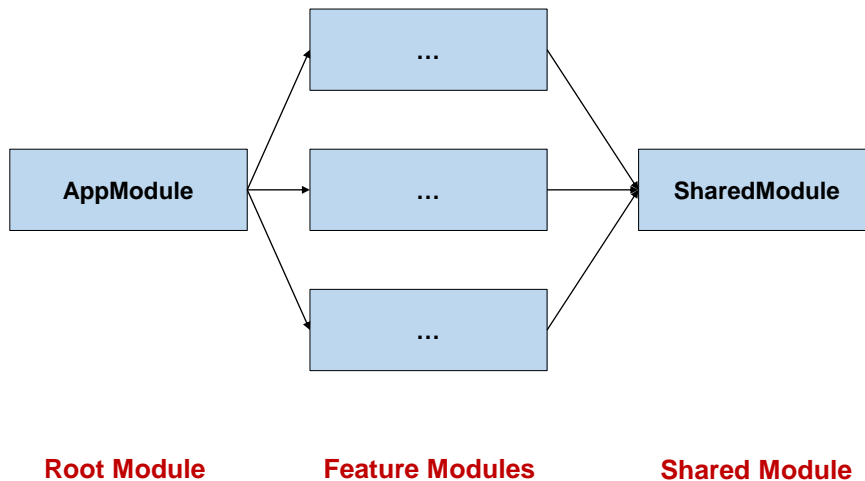


**Root Module**

**Feature Modules**

**Shared Module**

## Lazy Loading



Page • 11

SOFTWAREarchitekt.at

## Root Module with Lazy Loading

```

const APP_ROUTE_CONFIG: Routes = [
  {
    path: 'home',
    component: HomeComponent
  },
  {
    path: 'flights',
    loadChildren:
      './[...]/flight-booking.module#FlightBookingModule'
  }
];
  
```

Page • 12

SOFTWAREarchitekt.at

## Routes for "lazy" Module

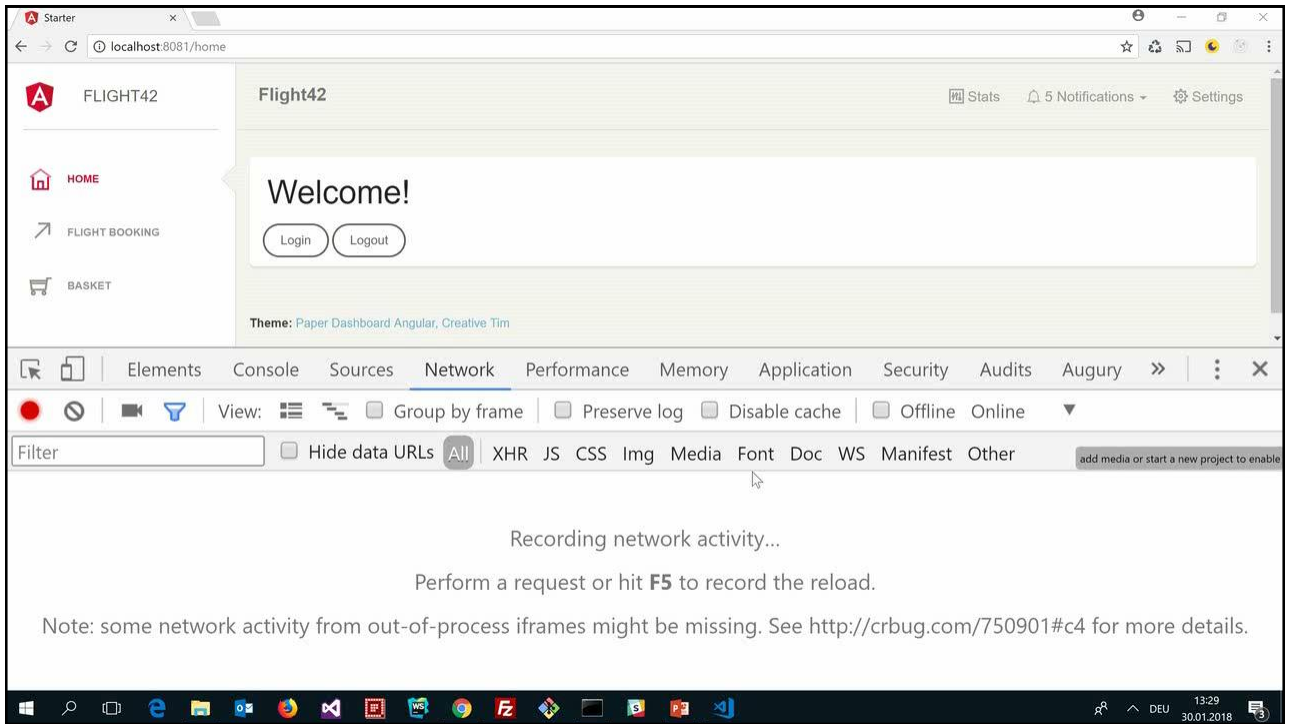
```
const FLIGHT_ROUTES = [
  {
    path: '',
    component: FlightBookingComponent,
    [...]
  },
  [...]
]
```

## Routes for "lazy" Module

```
const FLIGHT_ROUTES = [
  {
    path: 'subroute',
    component: FlightBookingComponent,
    [...]
  },
  [...]
]
```

flight-booking/subroute

Triggers Lazy Loading w/ loadChildren



## Lazy Loading

- Lazy Loading means: Loading it later
- Better startup performance
- Delay during execution for loading on demand

## Preloading



## Idea

- Module that might be needed later are loaded after the application started
- When module is needed it is available immediately



## Activate Preloading

```
...
imports: [
  [...]
  RouterModule.forRoot(
    ROUTE_CONFIG,
    { preloadingStrategy: PreloadAllModules });
]
...
```

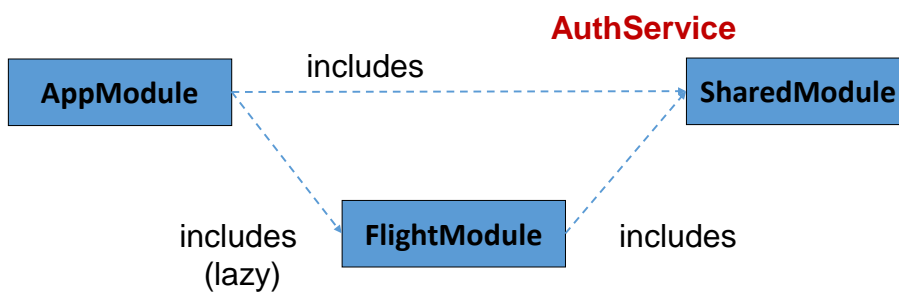
## Lazy Loading and Shared Modules



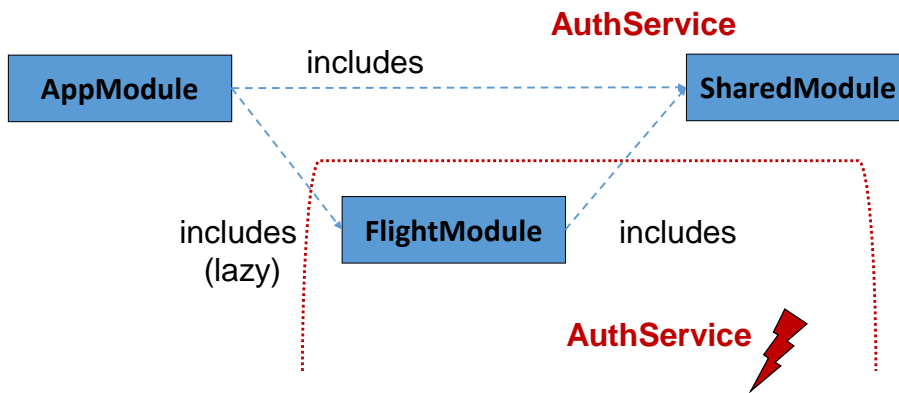


# DEMO

## Lazy Loading and Shared Modules



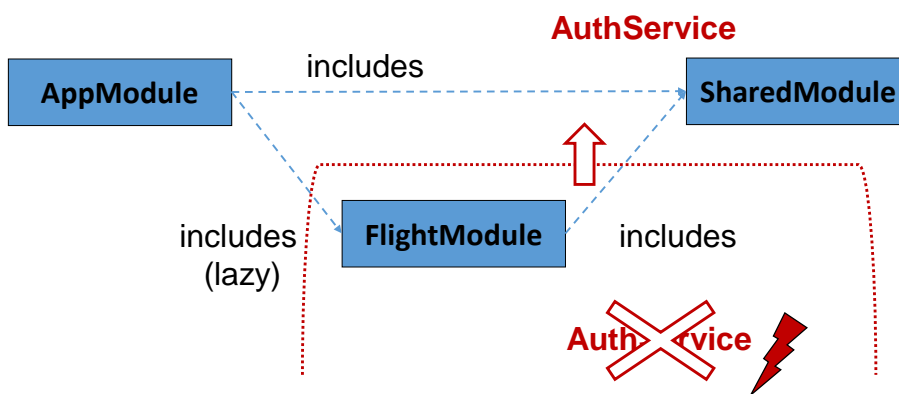
## Lazy Loading and Shared Modules



Page • 24

SOFTWAREarchitekt.at

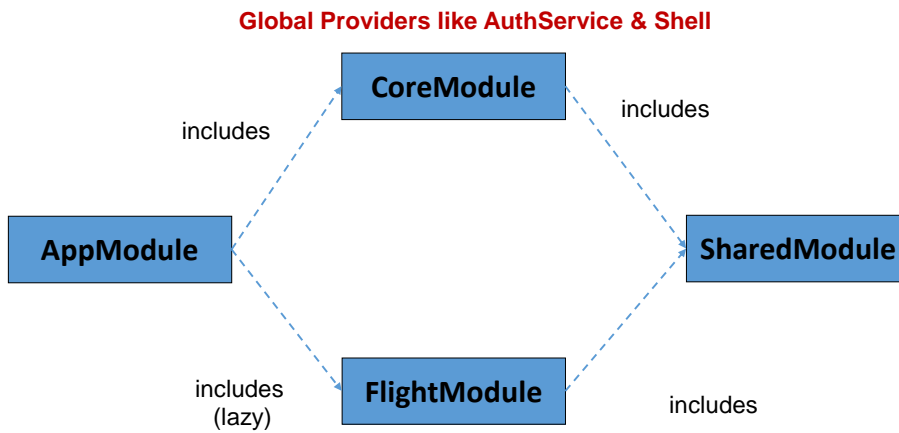
## Lazy Loading and Shared Modules



Page • 25

SOFTWAREarchitekt.at

## Solution



**Only import CoreModule into AppModule!**

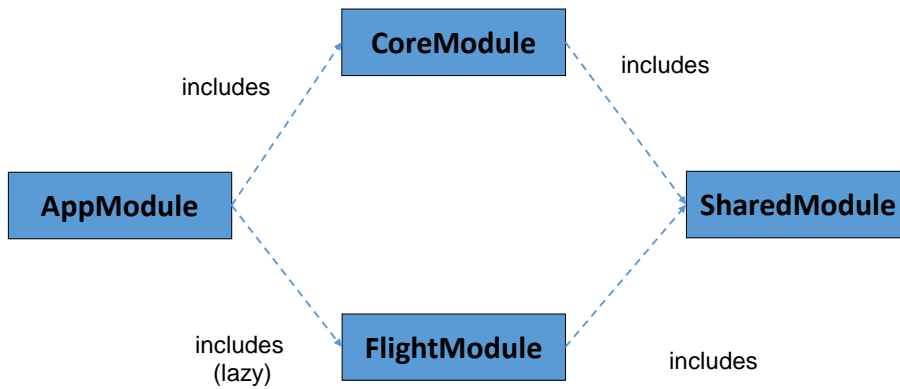
**SOFTWARE**architekt.at

Page • 26

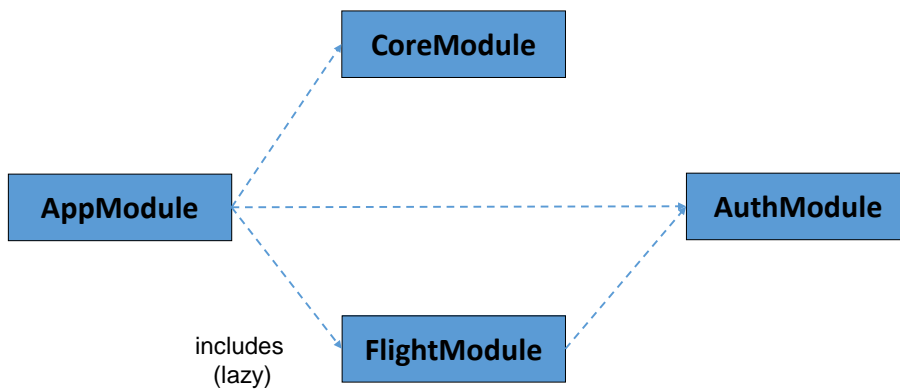
## DEMO

**SOFTWARE**architekt.at

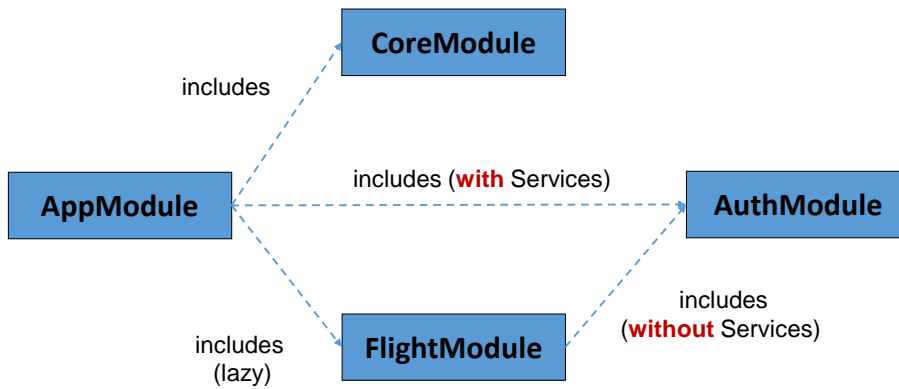
## Huge CoreModule?



## Solution (for Libraries)



## Solution (for Libraries)



## Auth Module

```

@NgModule({
  [...],
  providers: []
})
export class AuthModule {
}
  
```

## Auth Module

```
@NgModule({  
  [...],  
  providers: []  
})  
export class AuthModule {  
  static forRoot(): ModuleWithProviders {  
    return {  
      ngModule: AuthModule,  
      providers: [AuthService, [...]]  
    }  
  }  
}
```

## DEMO

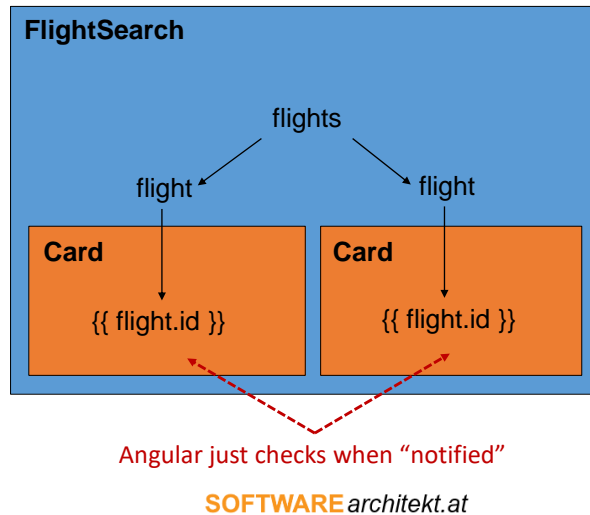


## Performance-Tuning with OnPush


DEMO



## OnPush



## "Notify" about change?

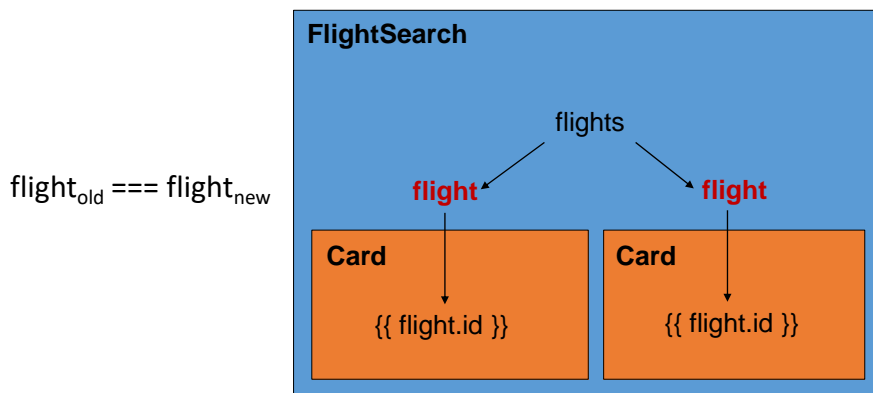
- Change bound data (@Input)
  - OnPush: Angular just compares the object reference!
  - e. g. `oldFlight === newFlight`
- Raise Event within the component
- Notify a bound observable 
- Trigger it manually
  - Don't do this at home ;-)
  - At least: Try to avoid this

## Activate OnPush

```
@Component({
  [...]
  changeDetection: ChangeDetectionStrategy.OnPush
})
export class FlightCard {
  [...]
  @Input() flight;
}
```

SOFTWAREarchitekt.at

## Change Inputs



SOFTWAREarchitekt.at

## Observables and OnPush

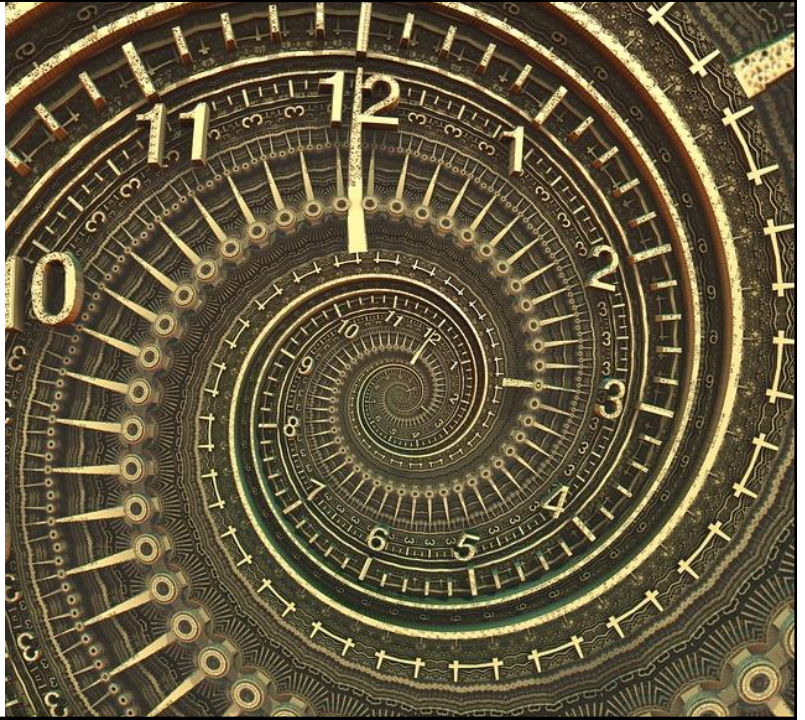
```
<flight-card  
  [item]="flight$ | async" [...]>  
</flight-card>
```

SOFTWAREarchitekt.at

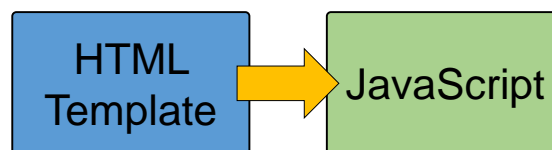
# DEMO

SOFTWAREarchitekt.at

Ahead of  
Time (AOT)  
Compilation



## Angular Compiler



Template Compiler

## Approaches

- JIT: Just in Time, at runtime
- AOT: Ahead of Time, during build

**SOFTWARE**architekt.at

## Advantages of AOT

- Better Startup-Performance
- Smaller Bundles: You don't need to include the compiler!
- Tools can easier analyse the code
  - Remove not needed parts of frameworks
  - Tree Shaking

**SOFTWARE**architekt.at

## Angular CLI

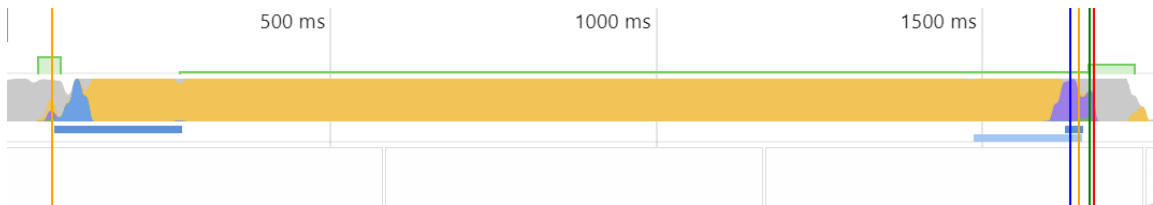
- `ng build --prod`
- `@ngtools/webpack` with `AotPlugin`
- Soon `AngularCompilerPlugin`
- Can be used without CLI too

**SOFTWARE***architekt.at*

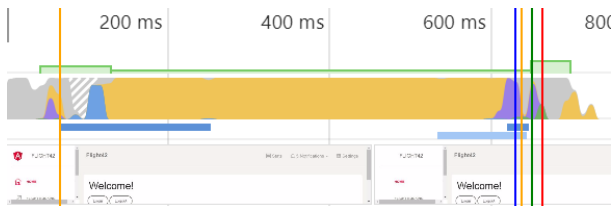
# DEMO

**SOFTWARE***architekt.at*

## Flight Search (Prod Build w/o AOT)



## Flight Search (Prod Build w/ AOT)





## Conclusion

Quick Wins

Lazy Loading  
and  
Preloading

OnPush w/  
Immutables and  
Observables

AOT and Tree  
Shaking

Use the CLI

**SOFTWARE**architekt.at