ANGULAR
ARCHITECTS
INSIDE KNOWLEDGE

Performance Tuning

Alex Thalhammer


Turbo Button
HIGHSCREEN
386 SX-33
POWER
HDD
TURBO
RESET

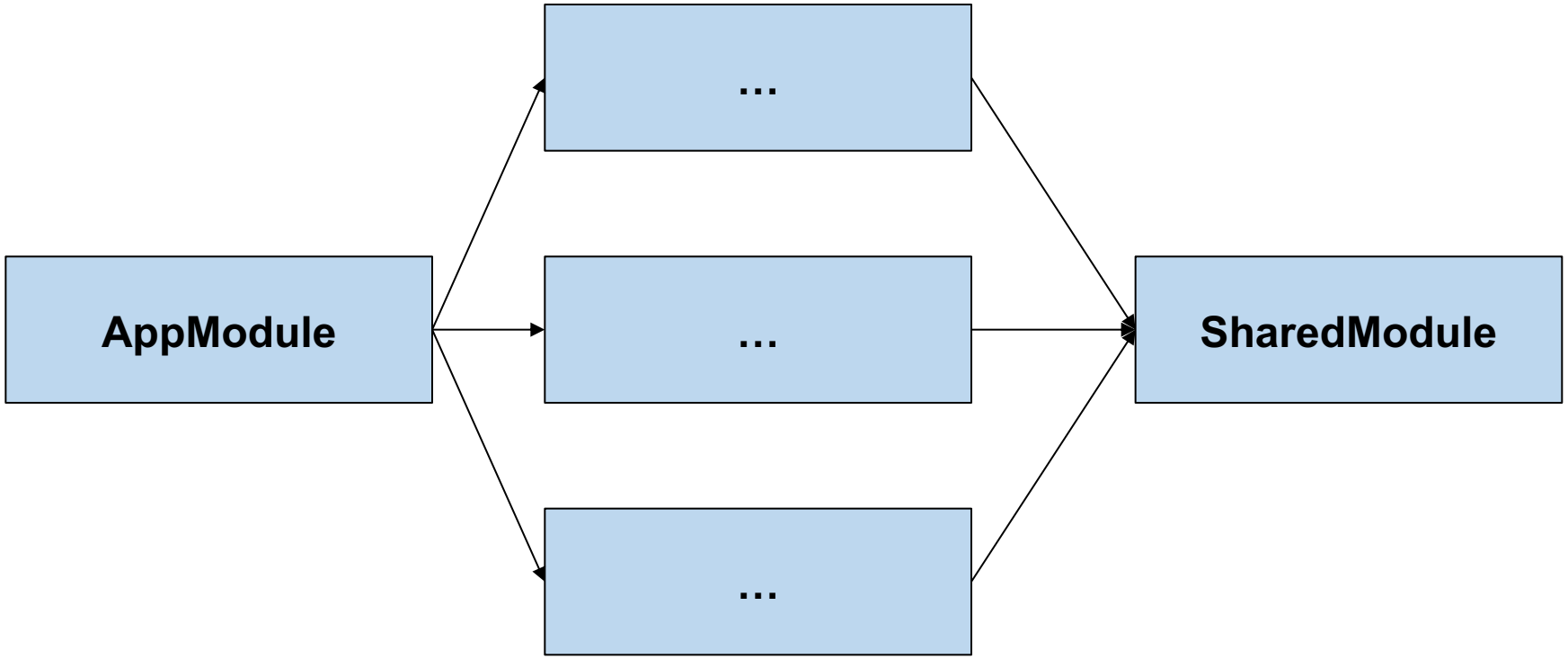
Contents

- Startup Performance
  - Lazy Loading and Preloading
  - (Manual) Build Analyzing & Optimization

- Runtime Performance
  - Data Binding with ChangeDetection.OnPush

Lazy Loading

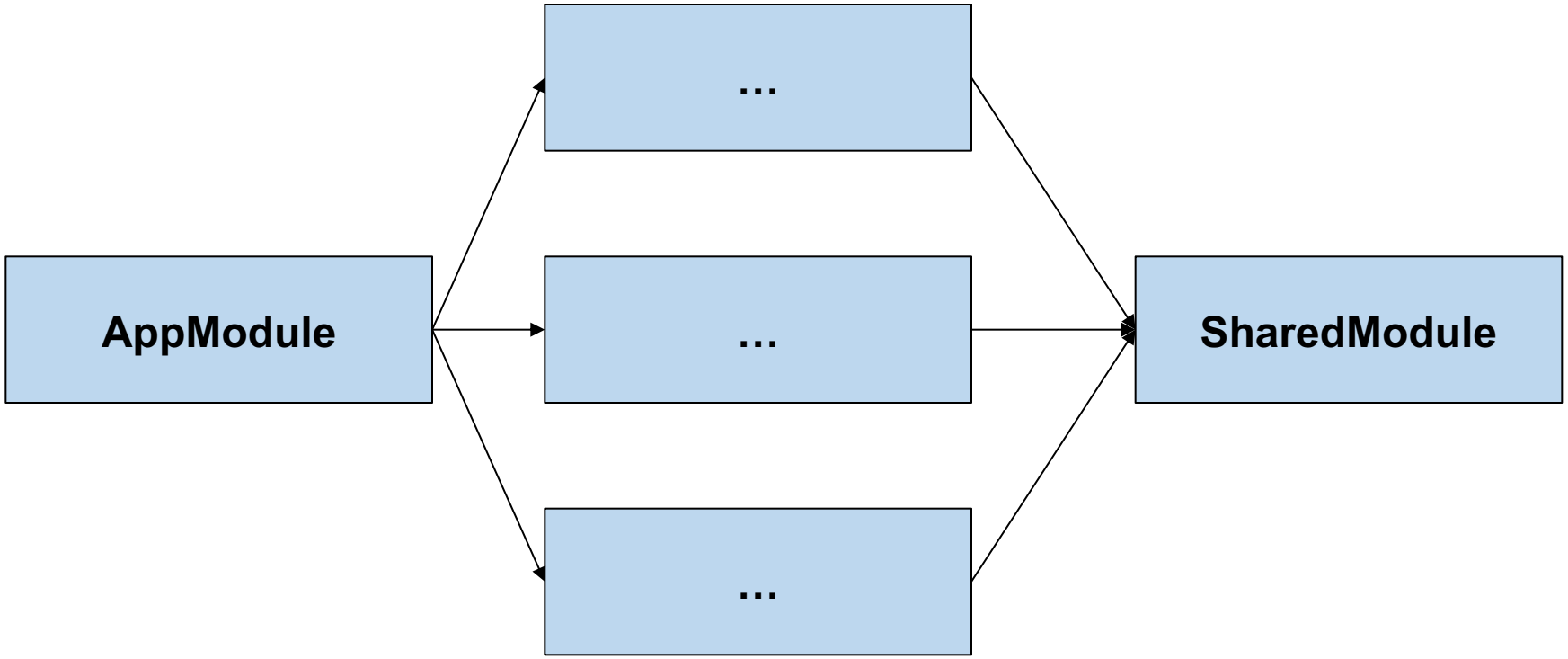
Module Structure



Root Module **Feature Modules** **Shared Module**

ANGULAR ARCHITECTS INSIDE KNOWLEDGE
SOFTWARE ARCHITECT

Lazy Loading



Root Module **Feature Modules** **Shared Module**

ANGULAR ARCHITECTS INSIDE KNOWLEDGE SOFTWARE ARCHITECT

Root Module with Lazy Loading

```
const APP_ROUTE_CONFIG: Routes = [
    {
        path: 'home',
        component: HomeComponent
    },
    {
        path: 'flight-booking',
        loadChildren: () => import('./flight-booking/flight-booking.module')
                              .then(m => m.FlightBookingModule)
    }
];
```


ANGULAR ARCHITECTS INSIDE KNOWLEDGE SOFTWARE ARCHITECT

Routes for "lazy" Module

```
const FLIGHT_ROUTES =    [
    {
        path: '',
        component: FlightSearchComponent,
        […]
    },
    […]
}
```


ANGULAR ARCHITECTS INSIDE KNOWLEDGE
SOFTWARE ARCHITECT

Routes for "lazy" Module

```
const FLIGHT_ROUTES =    [
    {
        path: ‘flight-search',
        component: FlightSearchComponent,
        […]
    },
    […]
}
```

flight-booking/flight-search

Triggers Lazy Loading w/ loadChildren

ANGULAR ARCHITECTS INSIDE KNOWLEDGE SOFTWARE ARCHITECT

DEMO

ANGULAR
ARCHITECTS
INSIDE KNOWLEDGE

SOFTWARE
ARCHITECT

Lazy Loading

- Lazy Loading means: Load it later, after startup
- Better initial load performance
- But: 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

ANGULAR ARCHITECTS INSIDE KNOWLEDGE
SOFTWARE ARCHITECT

Activate Preloading

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


ANGULAR ARCHITECTS INSIDE KNOWLEDGE
SOFTWARE ARCHITECT

Intelligent Preloading with ngx-quicklink

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

https://web.dev/route-preloading-in-angular/

https://www.npmjs.com/package/ngx-quicklink

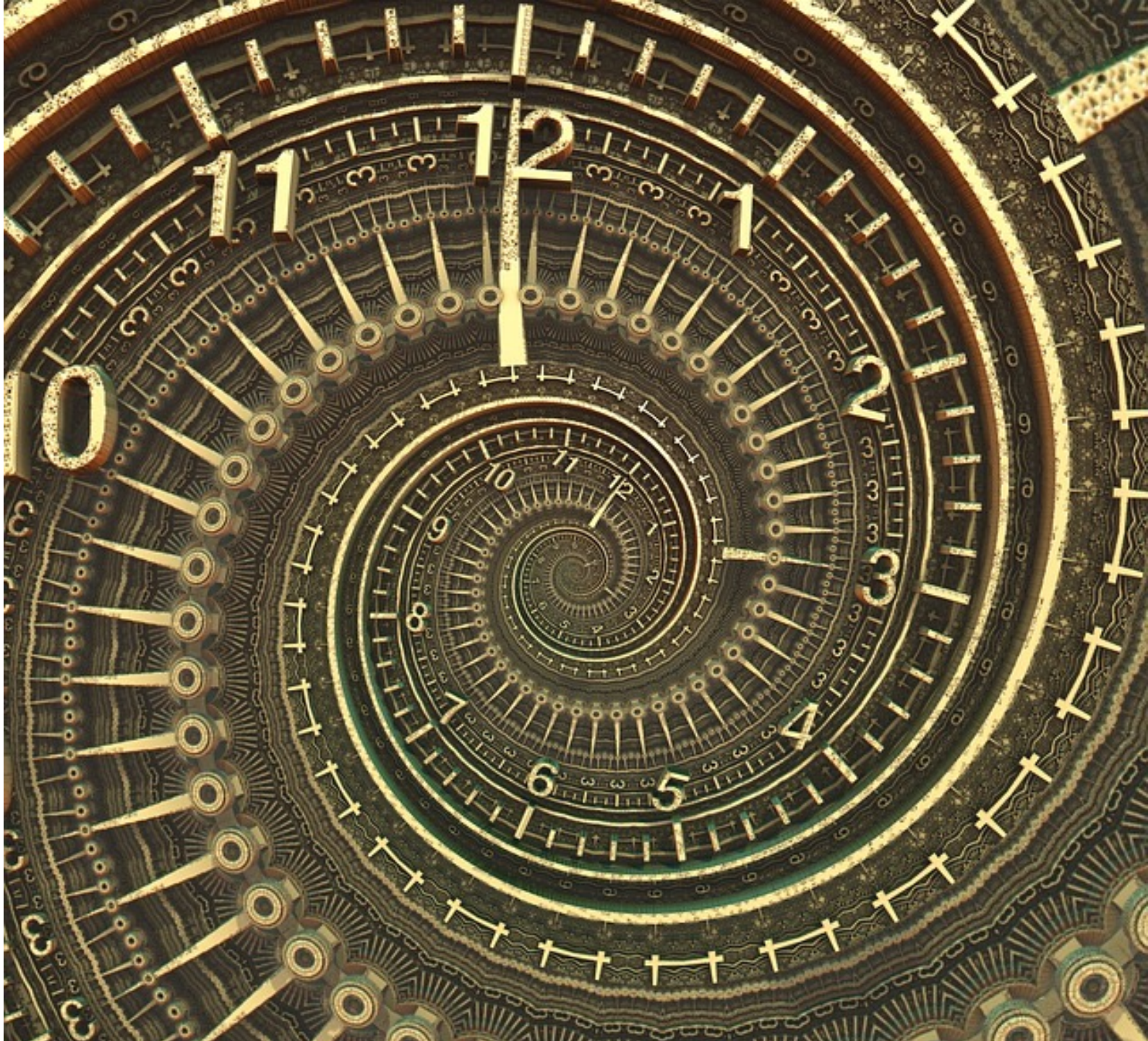
ANGULAR ARCHITECTS INSIDE KNOWLEDGE SOFTWARE ARCHITECT

Or CustomPreloadingStrategy

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

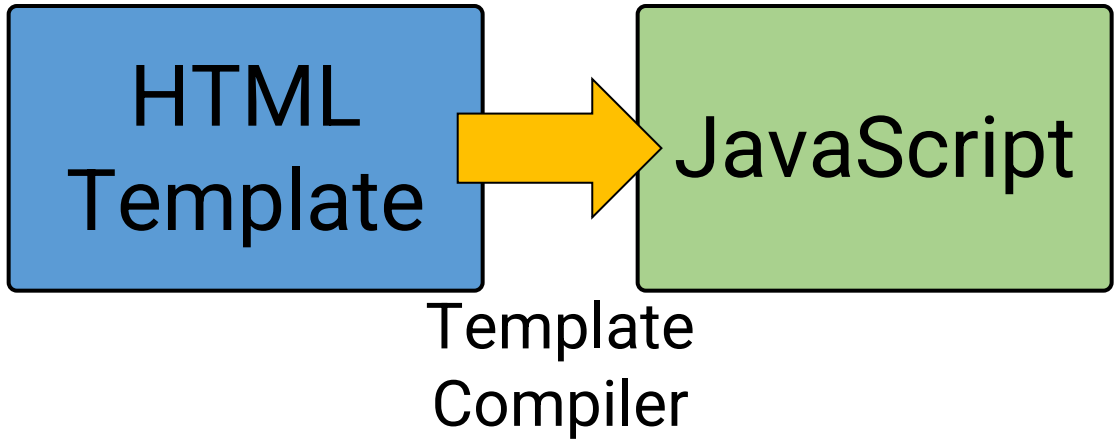

ANGULAR ARCHITECTS INSIDE KNOWLEDGE
SOFTWARE ARCHITECT

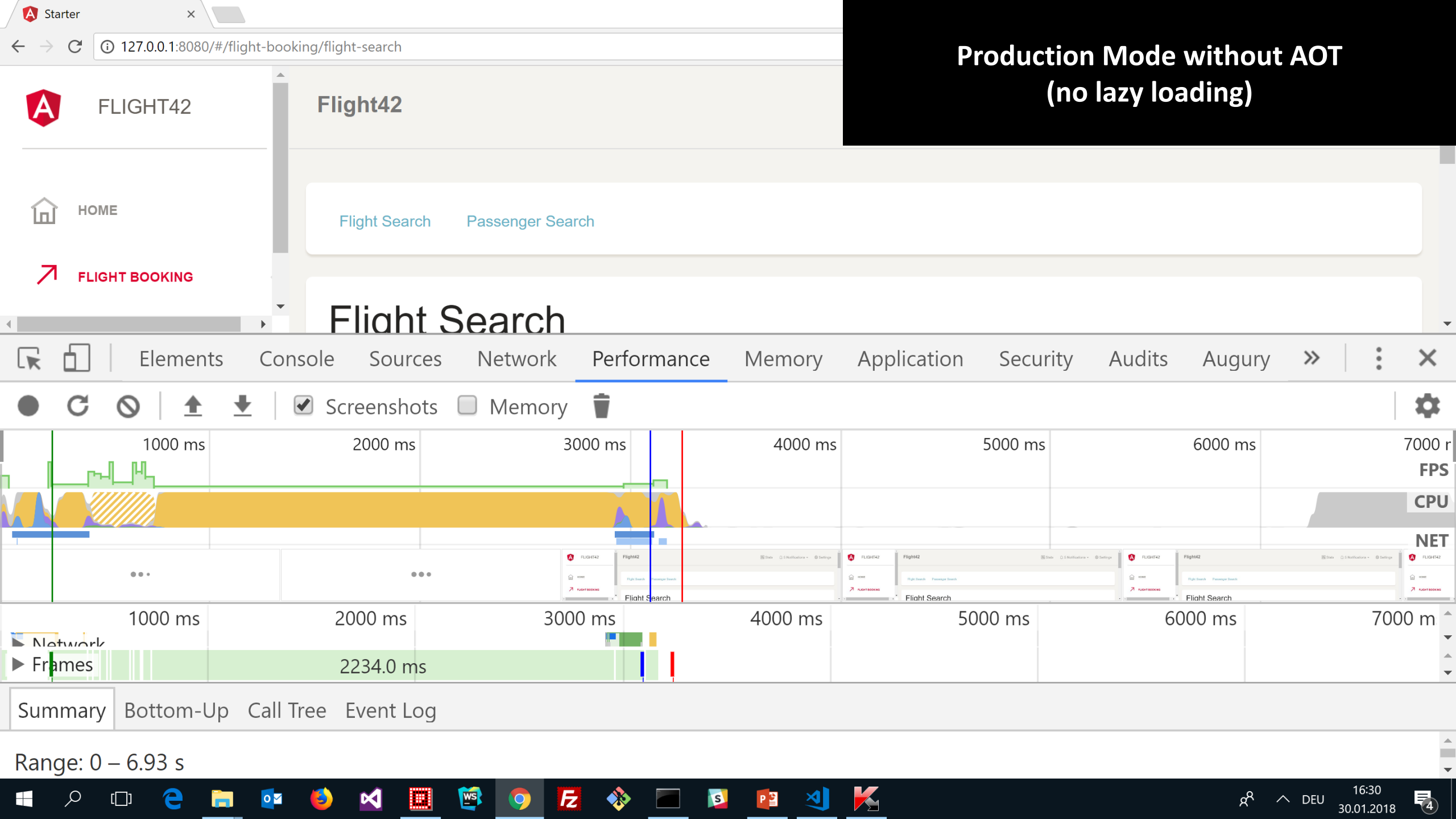
Ahead of Time (AOT) Compilation

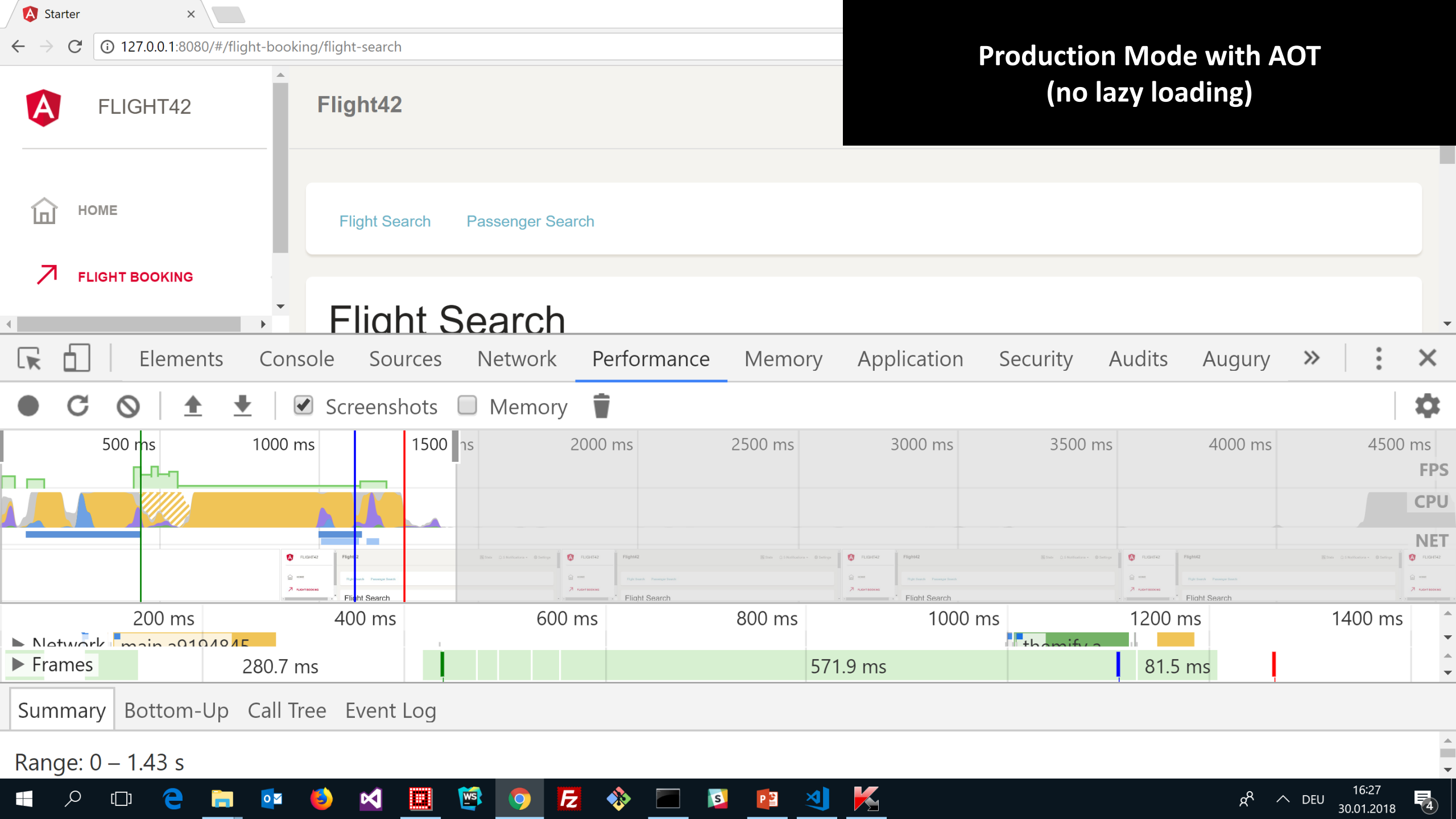


Approaches

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




Starter
127.0.0.1:8080/#/flight-booking/flight-search
Production Mode without AOT
(no lazy loading)
FLIGHT42
Flight42
HOME
FLIGHT BOOKING
Flight Search
Passenger Search
Flight Search
Elements
Console
Sources
Network
Performance
Memory
Application
Security
Audits
Augury
Screenshots
Memory
1000 ms
2000 ms
3000 ms
4000 ms
5000 ms
6000 ms
7000 ms
FPS
CPU
NET
Frames
2234.0 ms
Summary
Bottom-Up
Call Tree
Event Log
Range: 0 – 6.93 s
DEU
16:30
30.01.2018


Starter
127.0.0.1:8080/#/flight-booking/flight-search
Production Mode with AOT
(no lazy loading)
FLIGHT42
Flight42
HOME
FLIGHT BOOKING
Flight Search
Passenger Search
Flight Search
Elements
Console
Sources
Network
Performance
Memory
Application
Security
Audits
Augury
Screenshots
Memory
500 ms
1000 ms
1500
2000 ms
2500 ms
3000 ms
3500 ms
4000 ms
4500 ms
FPS
CPU
NET
200 ms
400 ms
600 ms
800 ms
1000 ms
1200 ms
1400 ms
Frames
280.7 ms
571.9 ms
81.5 ms
Summary
Bottom-Up
Call Tree
Event Log
Range: 0 – 1.43 s
DEU
16:27
30.01.2018

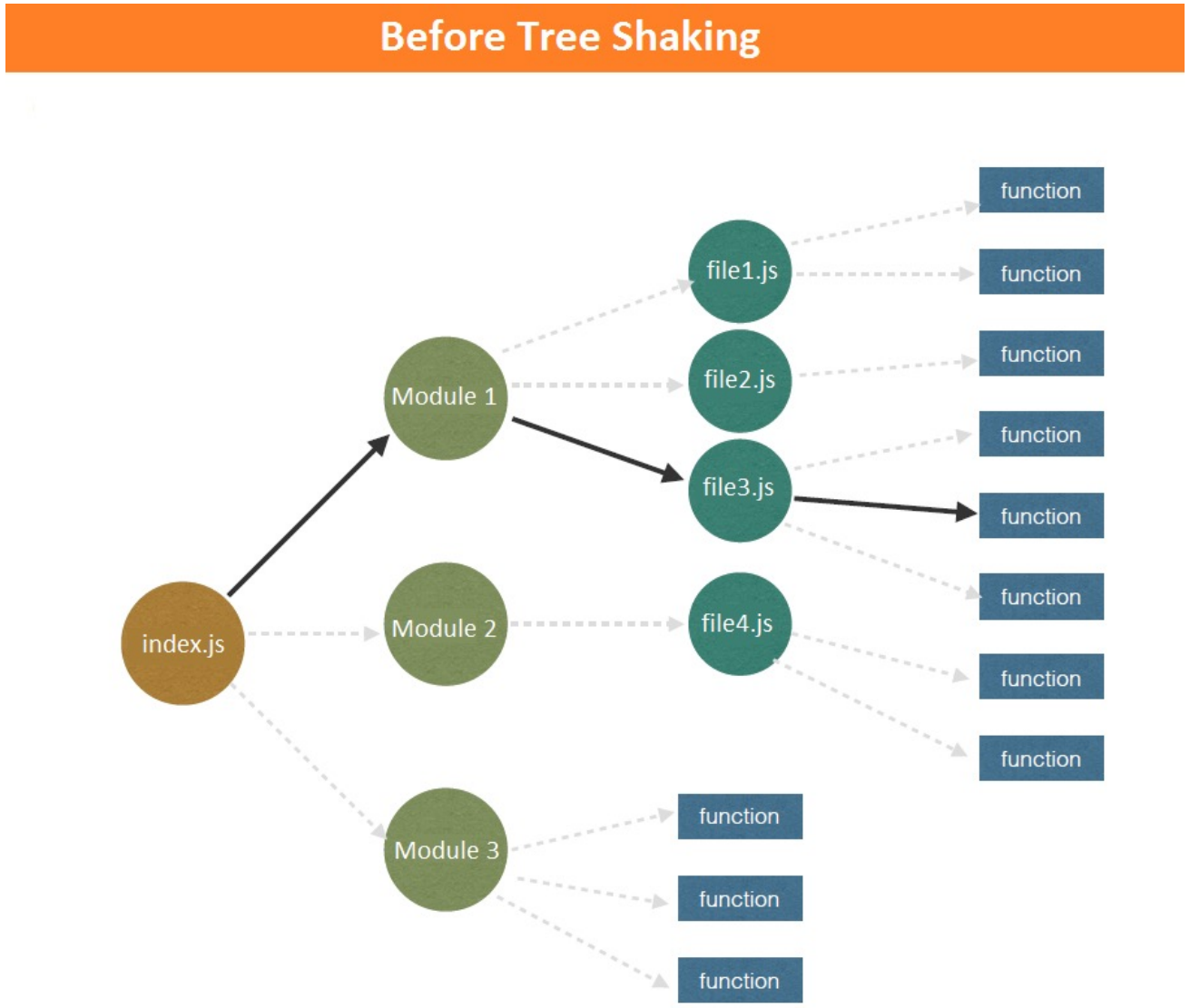
Advantages of AOT

- Ivy makes AOT the default ☺
  - Default for apps since NG 10
  - Default for libs default since NG 12

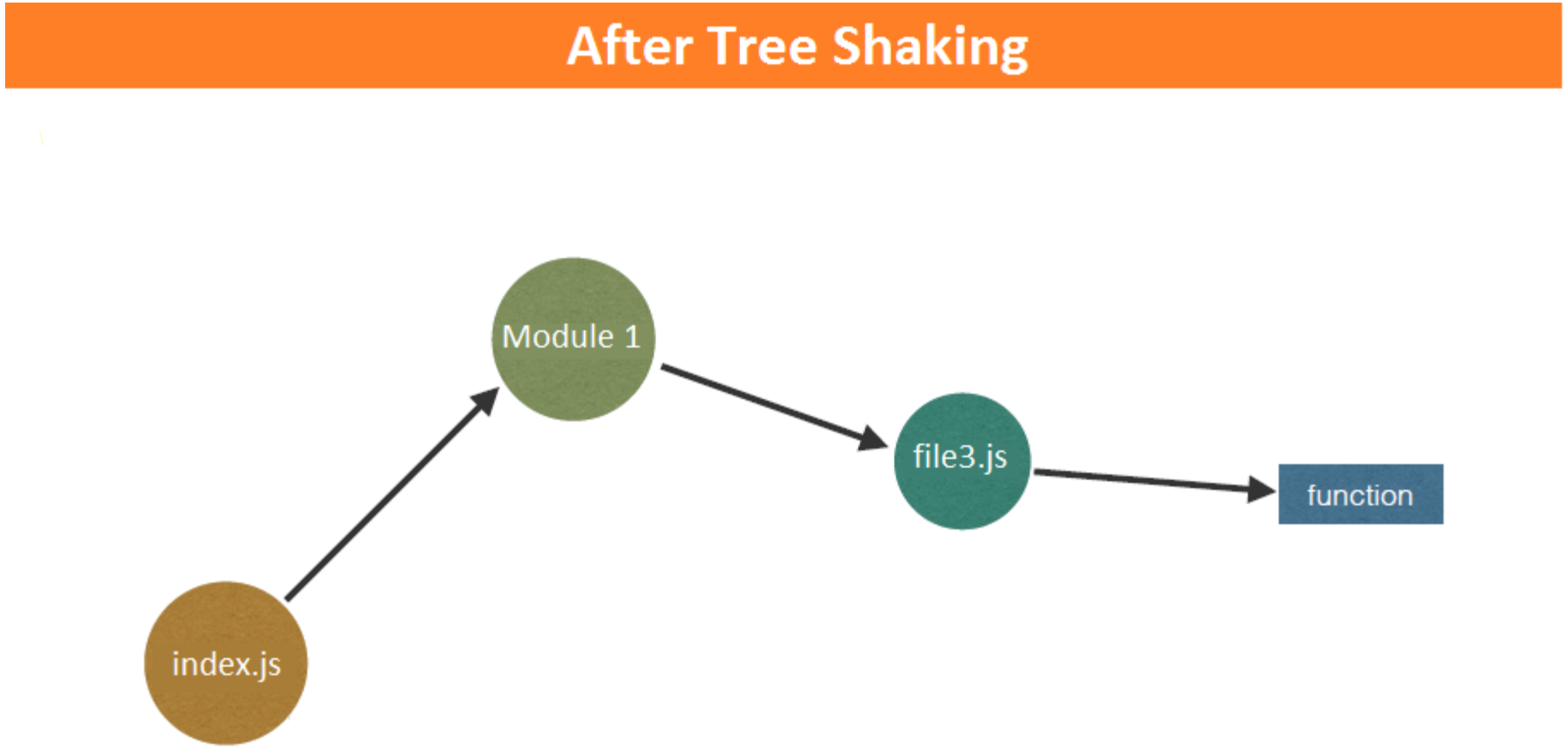
- Tools (e.g. Webpack) can easier analyse the code

- Smaller bundles → better Startup-Performance
  - You don't need to include the compiler!
  - **Tree Shaking**: Remove unused parts of framework & libs

Tree Shaking



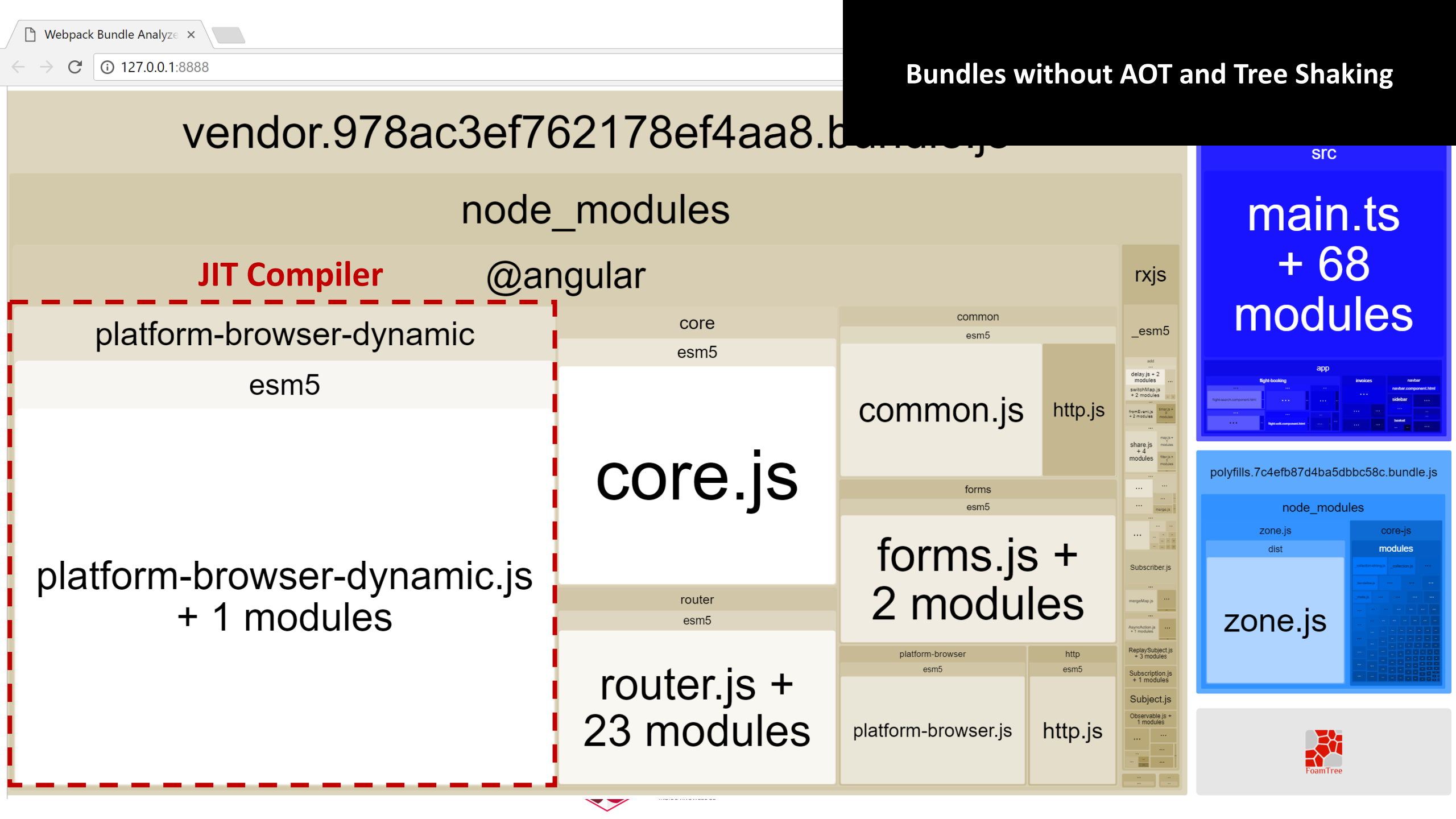
Tree Shaking



Webpack Bundle Analyzer

ANGULAR
ARCHITECTS
INSIDE KNOWLEDGE

SOFTWARE
ARCHITECT


Webpack Bundle Analyze
127.0.0.1:8888
Bundles without AOT and Tree Shaking
vendor.978ac3ef762178ef4aa8.b
node_modules
JIT Compiler
@angular
platform-browser-dynamic
esm5
platform-browser-dynamic.js
+ 1 modules
core
esm5
core.js
router
esm5
router.js +
23 modules
common
esm5
common.js
http.js
forms
esm5
forms.js +
2 modules
platform-browser
esm5
platform-browser.js
http
esm5
http.js
rxjs
_esm5
Subscriber.js
Subject.js
src
main.ts
+ 68
modules
app
polyfills.7c4efb87d4ba5dbbc58c.bundle.js
node_modules
zone.js
dist
zone.js
core-js
modules
FoamTree

Demo

ANGULAR
ARCHITECTS
INSIDE KNOWLEDGE

SOFTWARE
ARCHITECT

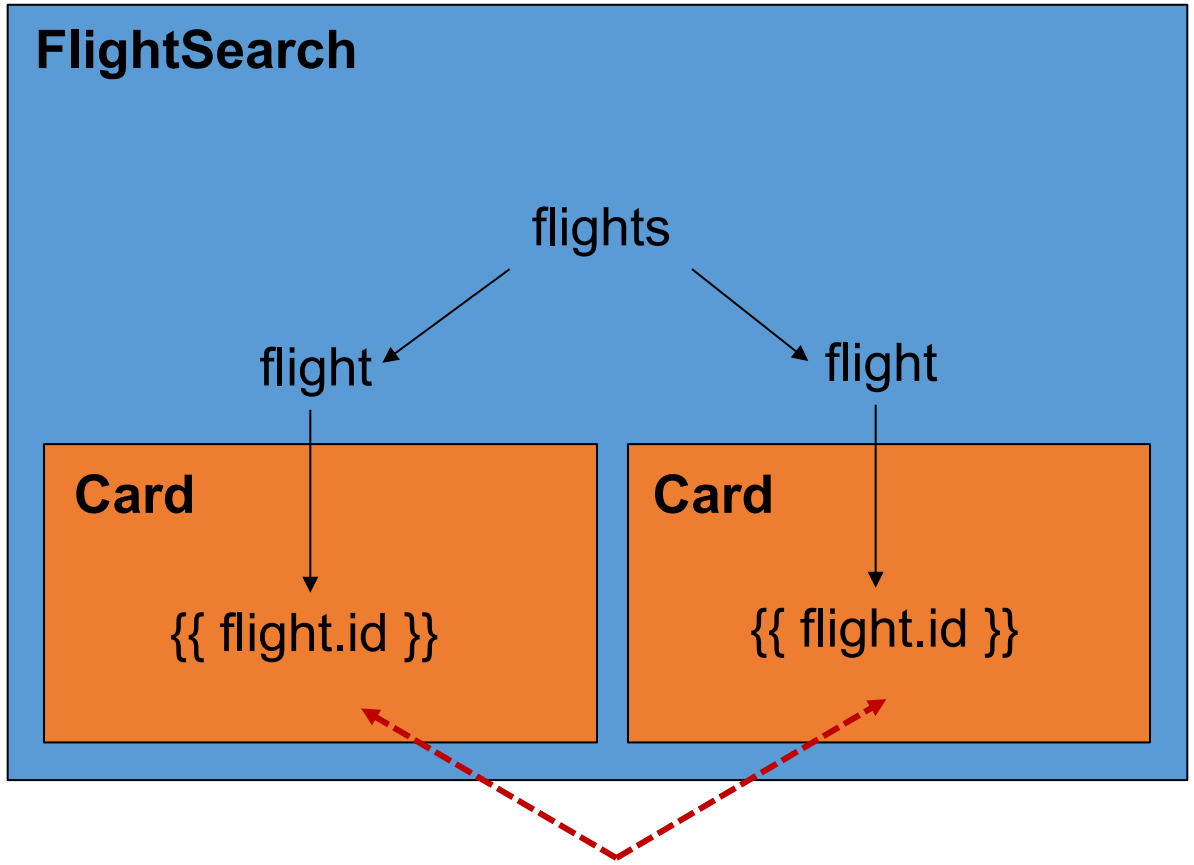
Performance-Tuning with OnPush

DEMO

ANGULAR
ARCHITECTS
INSIDE KNOWLEDGE

SOFTWARE
ARCHITECT

OnPush



Angular just checks when “notified”

"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
  - {{ flights$ | async }}
- Trigger it manually w ChangeDetectorReference
  - Don't do this at home ;-)
  - At least: Try to avoid this

Activate OnPush

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

DEMO

ANGULAR
ARCHITECTS
INSIDE KNOWLEDGE

SOFTWARE
ARCHITECT

Conclusion

Quick Wins

Lazy Loading & Preloading

Manual Build Optimization

OnPush w/ Immutables and Observables

For performance deep dive

Watch this (starting at 8:30):

https://drive.google.com/file/d/15fmyedJPYSOIv_0YvFtg26XGS8tZpZ03/view

Repo: https://github.com/jeffbcross/victor-videos/

ANGULAR ARCHITECTS INSIDE KNOWLEDGE
SOFTWARE ARCHITECT