ANGULAR
ARCHITECTS
INSIDE KNOWLEDGE

Performance Tuning

Alex Thalhammer.


Turbo Button
HIGHSCREEN
386 SX-33
POWER
HDD
TURBO
RESET

Contents

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

Lazy Loading

Why Lazy Loading?

- Load modules when they are needed

- Improve initial load (performance → very important!)

ANGULAR ARCHITECTS INSIDE KNOWLEDGE
SOFTWARE ARCHITECT

Root module with Lazy Loading

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

    }
];
```

Routes for feature module

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

```
export const FlightRouterModule =
                    RouterModule.forChild(FLIGHT_ROUTES);
```


ANGULAR ARCHITECTS INSIDE KNOWLEDGE
SOFTWARE ARCHITECT

DEMO

ANGULAR ARCHITECTS
INSIDE KNOWLEDGE
SOFTWARE ARCHITECT

Preloading

Idea

- Once the initial load (the important one) is complete load the lazy loaded modules (before they are even used)

- Once the module will come into use it's immediately accessable

ANGULAR ARCHITECTS INSIDE KNOWLEDGE SOFTWARE ARCHITECT

Use preloading (very easy!)

```
export const AppRoutesModule =
                  RouterModule.forRoot(
                    ROUTE_CONFIG,
                    { preloadingStrategy: PreloadAllModules });
```


ANGULAR ARCHITECTS INSIDE KNOWLEDGE
SOFTWARE ARCHITECT

DEMO

ANGULAR ARCHITECTS INSIDE KNOWLEDGE
SOFTWARE ARCHITECT

LAB

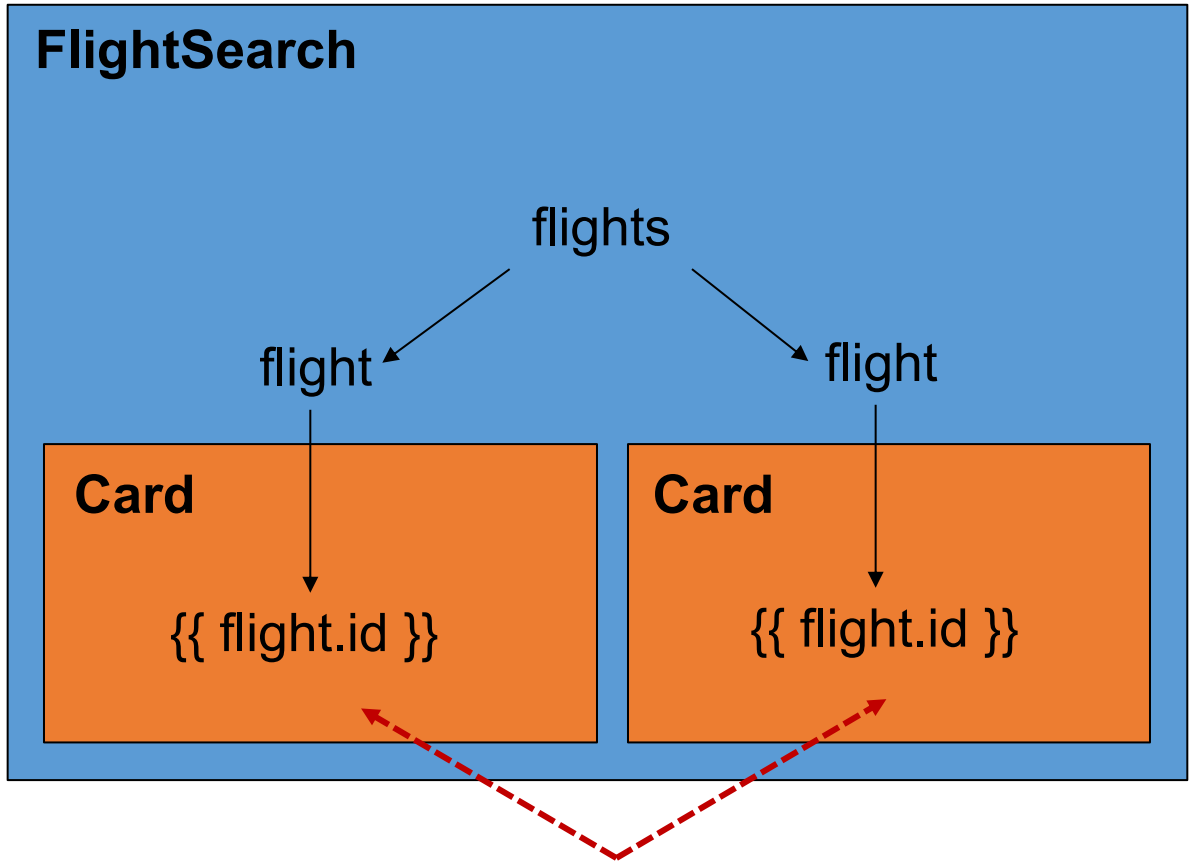
ANGULAR ARCHITECTS
INSIDE KNOWLEDGE
SOFTWARE ARCHITECT

Performance-Tuning with OnPush

DEMO

ANGULAR ARCHITECTS INSIDE KNOWLEDGE
SOFTWARE ARCHITECT

OnPush



Angular just checks when “notified”

ANGULAR ARCHITECTS INSIDE KNOWLEDGE
SOFTWARE ARCHITECT

"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
  - Don't do this at home ;-)
  - At least: Try to avoid this

ANGULAR ARCHITECTS INSIDE KNOWLEDGE SOFTWARE ARCHITECT

Activate OnPush

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


ANGULAR ARCHITECTS INSIDE KNOWLEDGE
SOFTWARE ARCHITECT

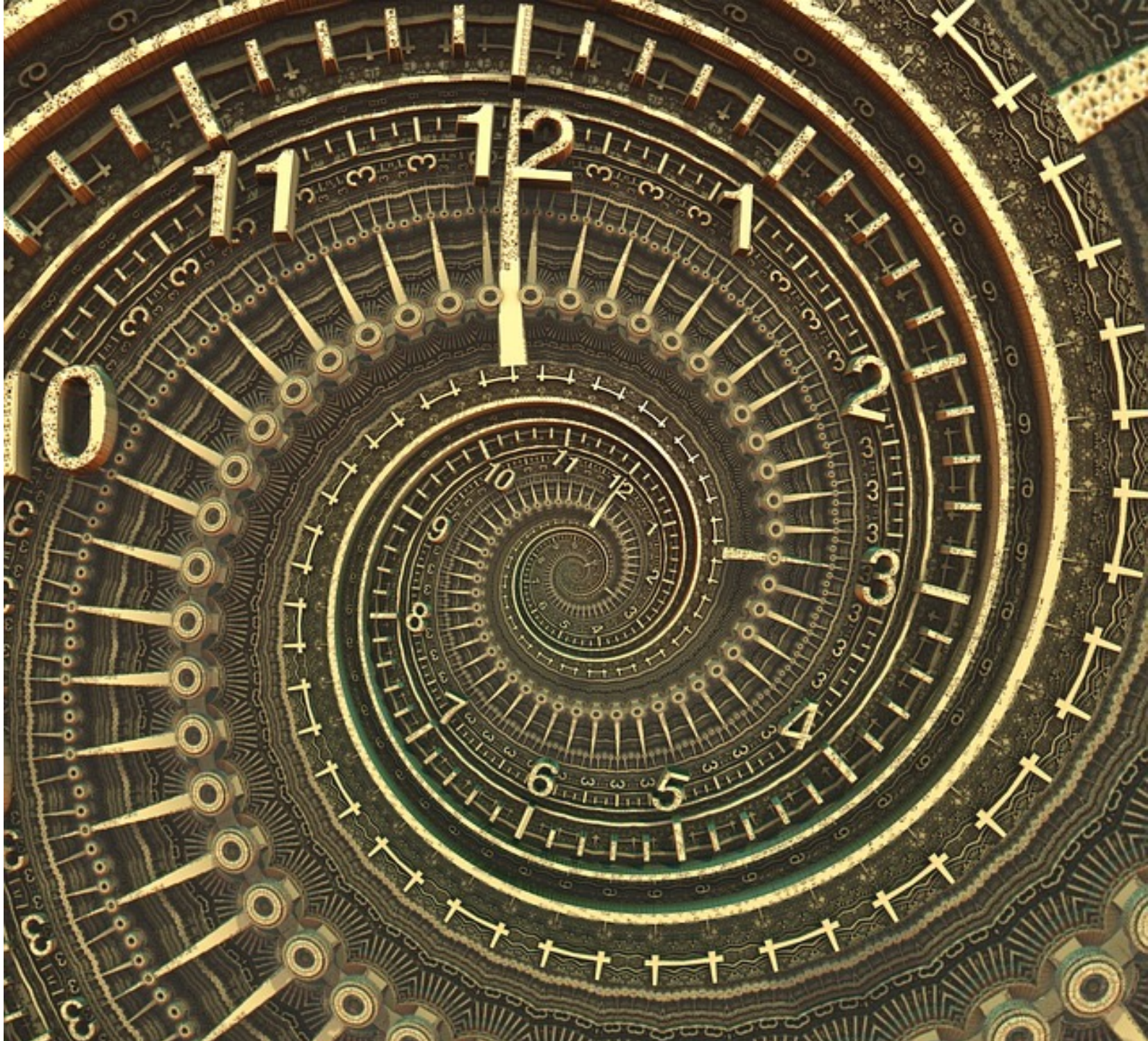
DEMO

ANGULAR ARCHITECTS INSIDE KNOWLEDGE
SOFTWARE ARCHITECT

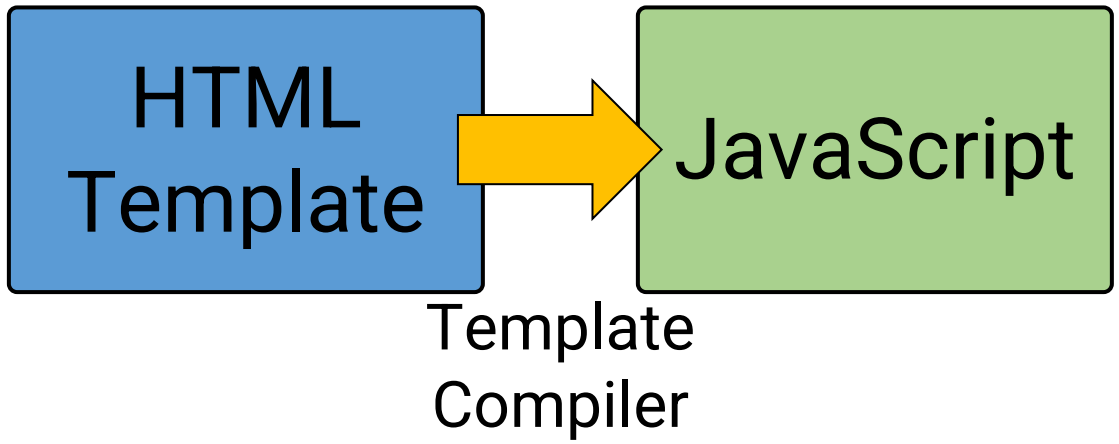
LAB

ANGULAR ARCHITECTS INSIDE KNOWLEDGE
SOFTWARE ARCHITECT

Ahead of Time (AOT) Compilation



Angular Compiler



ANGULAR ARCHITECTS
INSIDE KNOWLEDGE
SOFTWARE ARCHITECT

Approaches

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

ANGULAR ARCHITECTS INSIDE KNOWLEDGE
SOFTWARE ARCHITECT

Advantages of AOT

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

ANGULAR ARCHITECTS INSIDE KNOWLEDGE SOFTWARE ARCHITECT

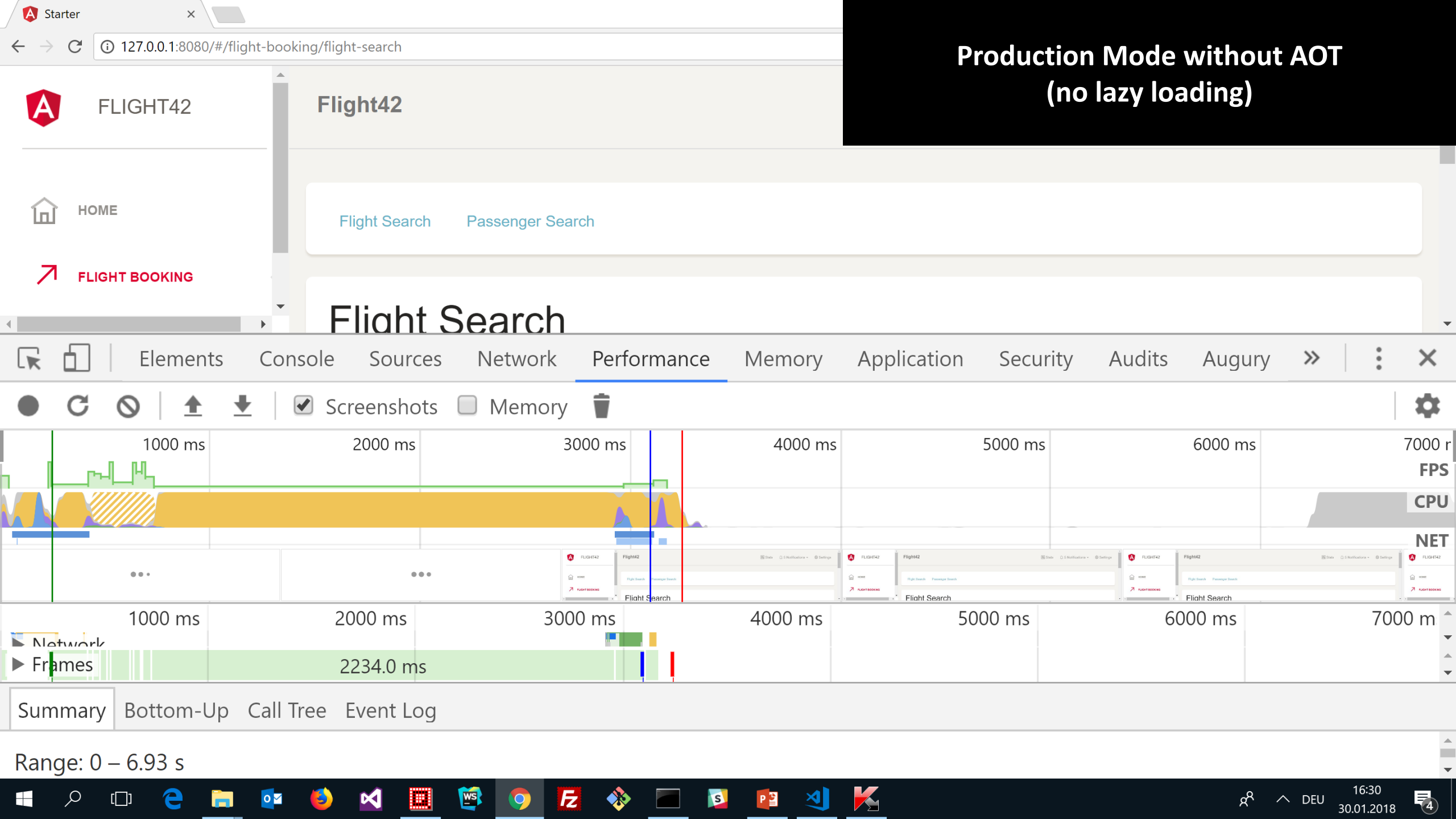
Angular CLI

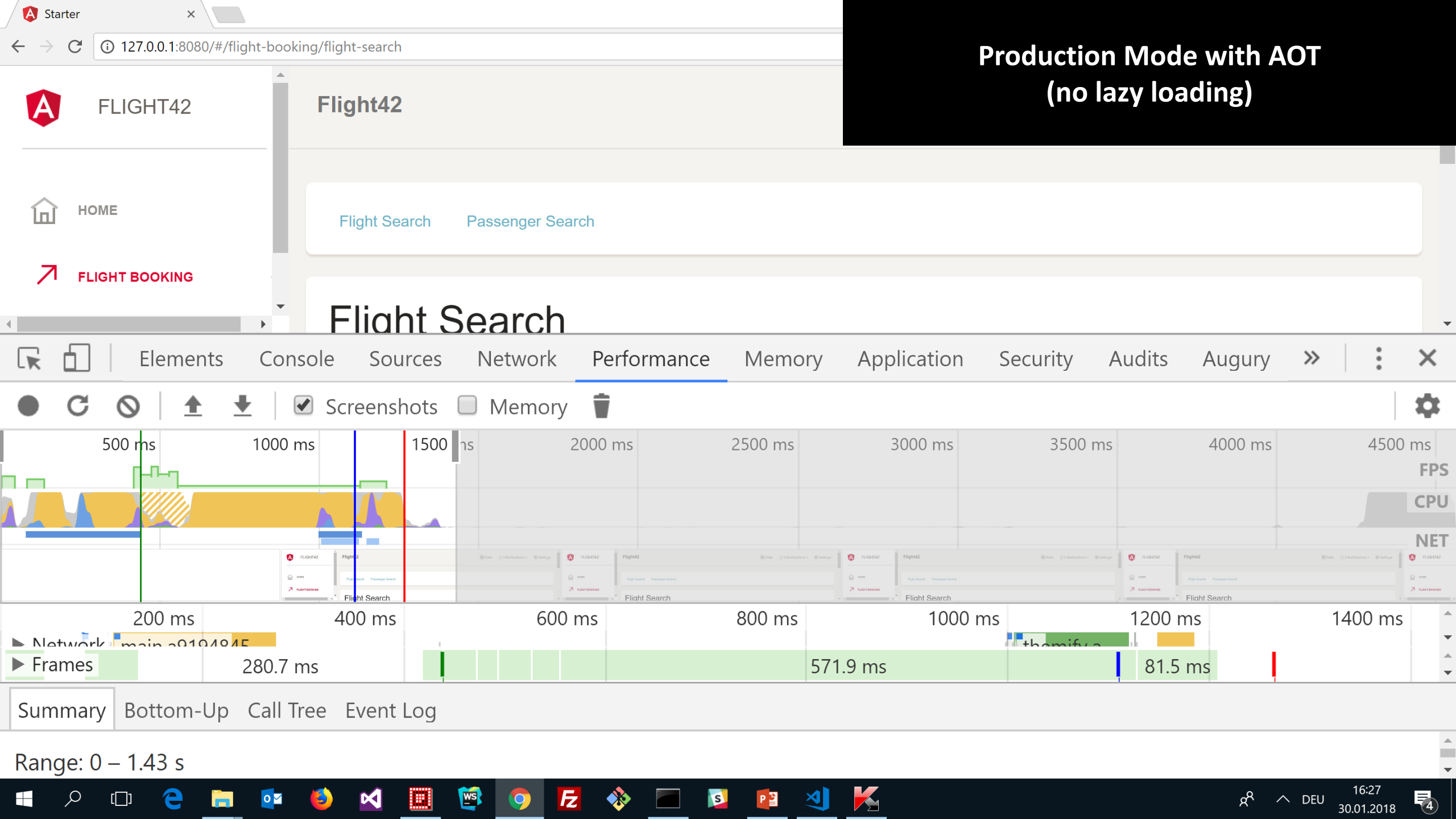
- ng build --prod
- @ngtools/webpack with AngularCompilerPlugin
- Can be used without CLI too

ANGULAR ARCHITECTS INSIDE KNOWLEDGE
SOFTWARE ARCHITECT

DEMO

ANGULAR ARCHITECTS
INSIDE KNOWLEDGE
SOFTWARE ARCHITECT


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

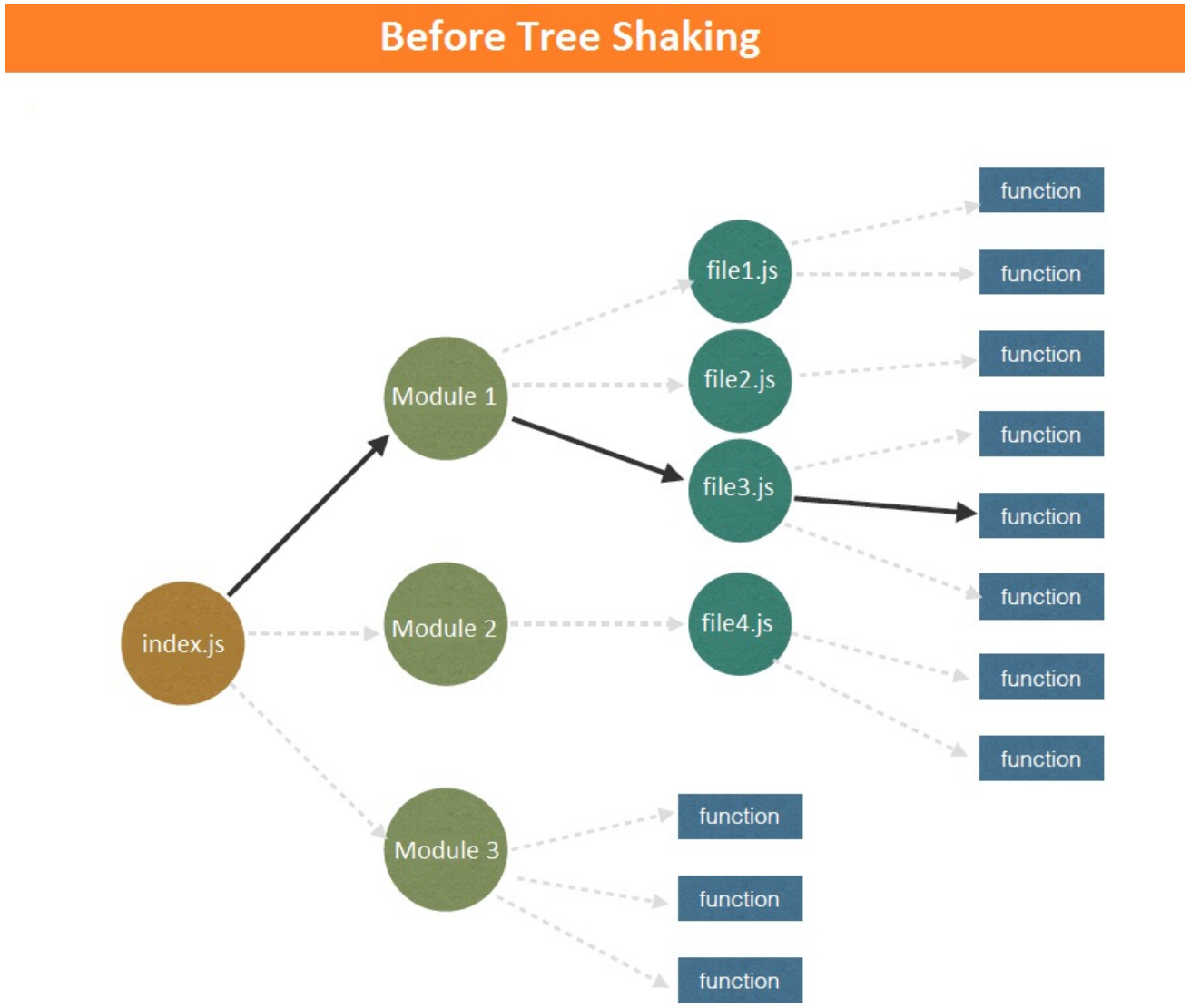

Starter
127.0.0.1:8080/#/flight-booking/flight-search
Production Mode with AOT
(no lazy loading)
FLIGHT42
Flight42
HOME
FLIGHT BOOKING
Flight Search
Passenger Search
Flight Search
Elements
Console
Sources
Network
Performance
Memory
Application
Security
Audits
Augury
Screenshots
Memory
500 ms
1000 ms
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

Ivy makes AOT the default ☺

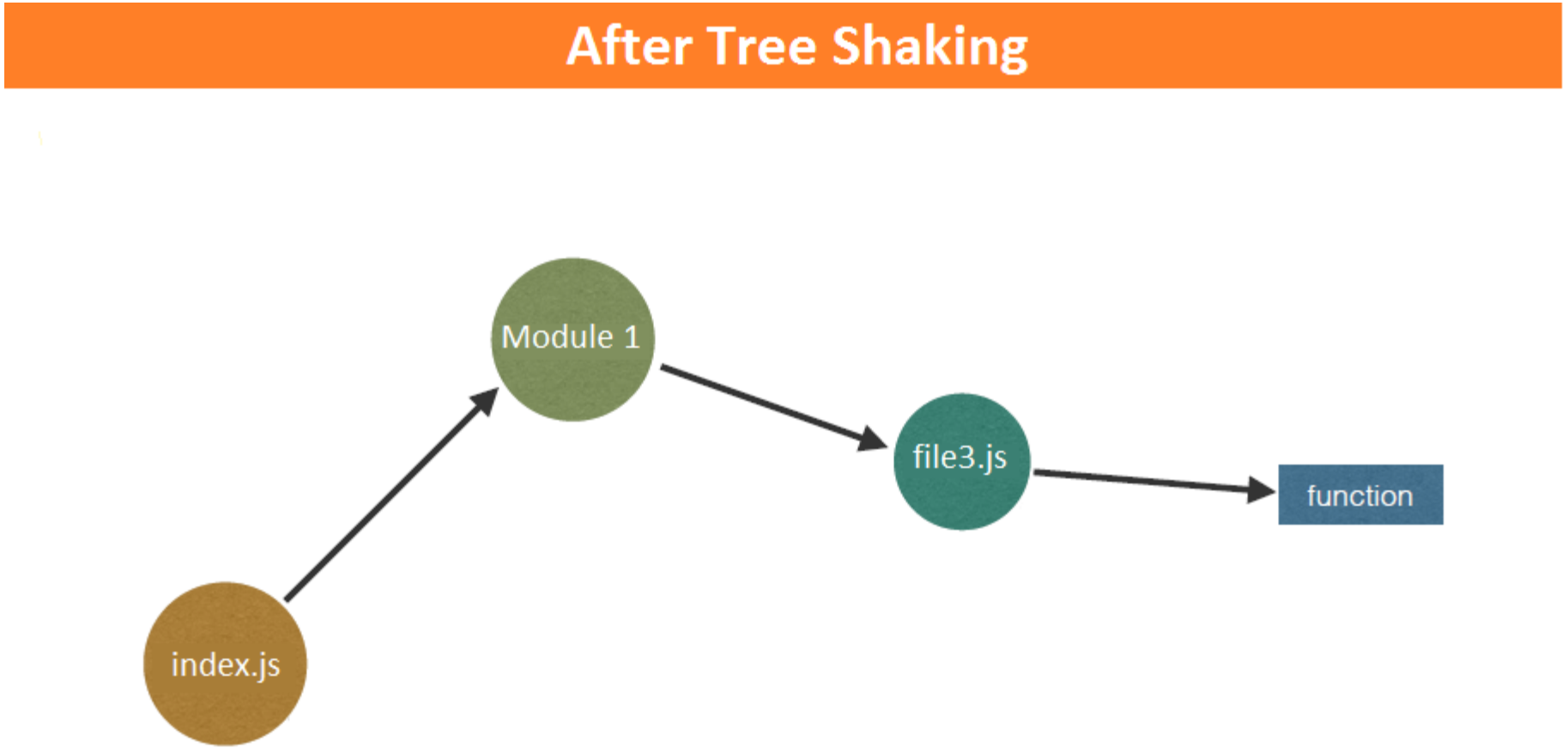
- Ivy also does a lot of under the hood optimization
- No breaking changes, nothing to do from our side ☺
- Angular ViewEngine itself was not tree-shakable
- Angular Ivy is tree-shakable ☺
- Default since NG 10, for libs default since NG 12

ANGULAR ARCHITECTS INSIDE KNOWLEDGE SOFTWARE ARCHITECT

Tree Shaking



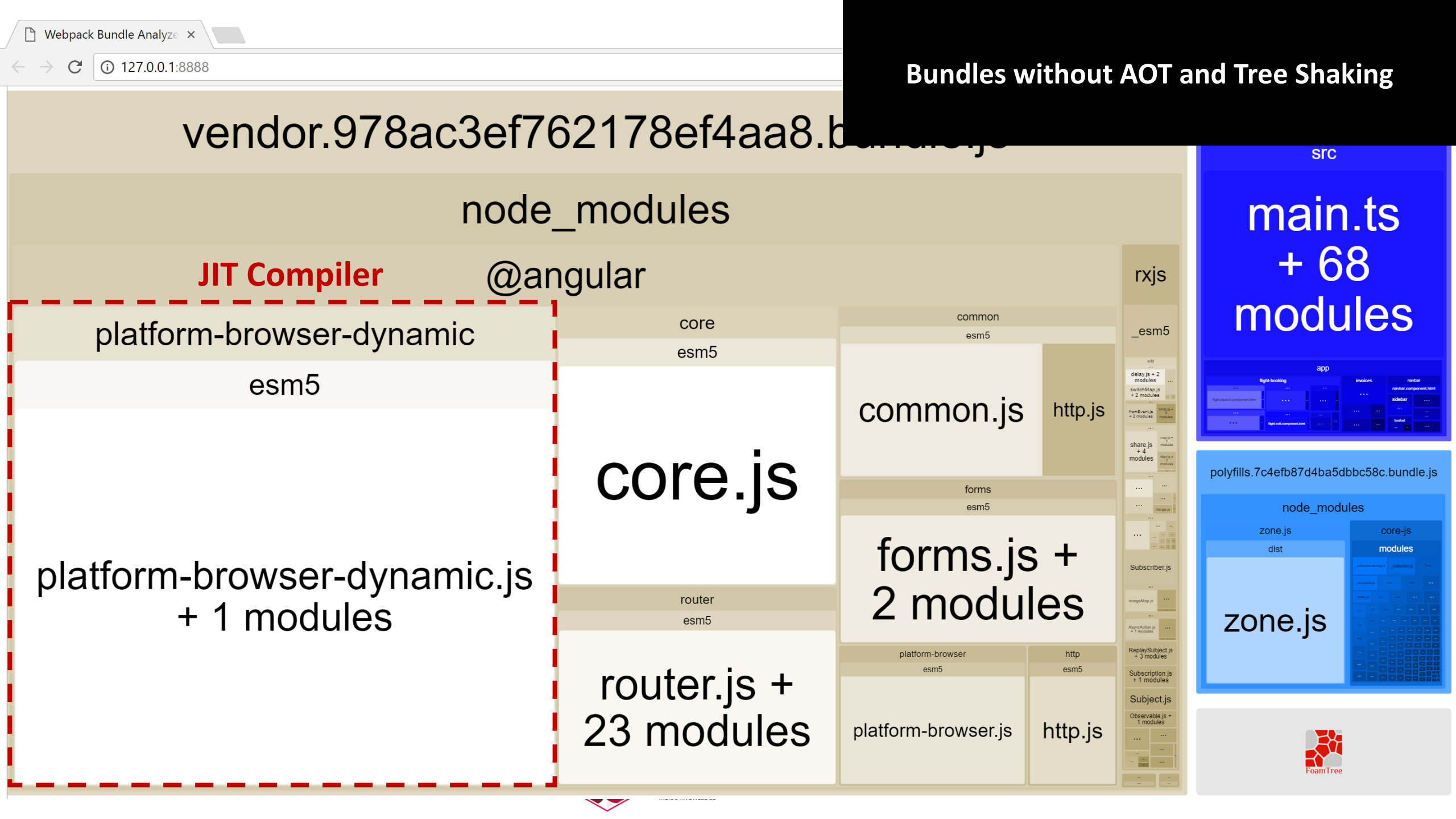
Tree Shaking



ANGULAR ARCHITECTS INSIDE KNOWLEDGE
SOFTWARE ARCHITECT

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

Conclusion

Lazy Loading

Preloading

OnPush w/ Immutables and Observables

AOT and Tree Shaking

ANGULAR ARCHITECTS INSIDE KNOWLEDGE
SOFTWARE ARCHITECT

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