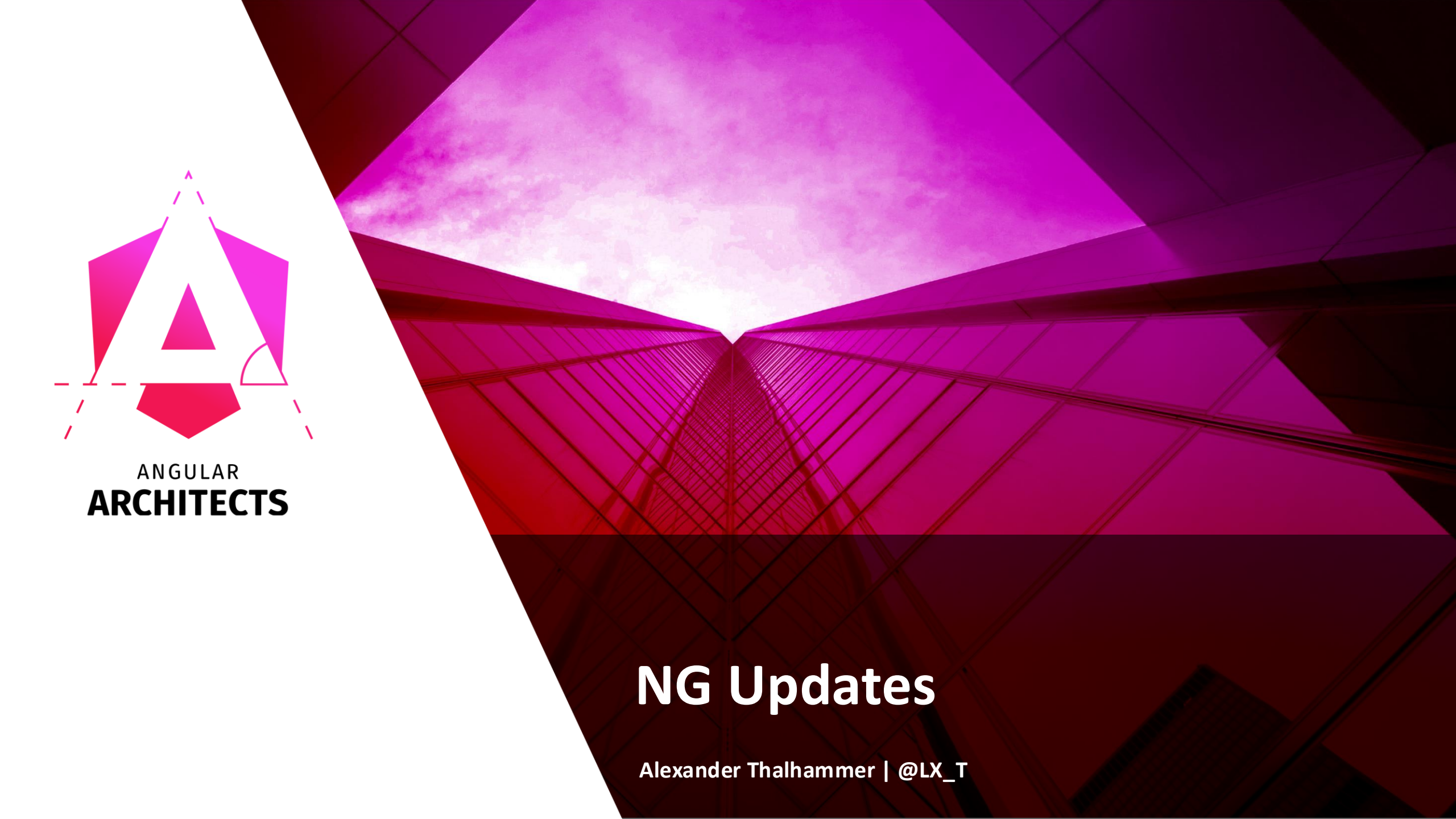


ANGULAR
ARCHITECTS

NG Updates

Alexander Thalhammer | @LX_T

NG Updates

- 14: Typed Reactive Forms, NgOptimizedImage
- 15: Standalone Components (migration!)
- 16: Signals
  - takeUntilDestroyed() operator (& DestroyRef)
  - SSR: Non Destructive Hydration
  - required @Input()
  - Vite & esbuild
  - withComponentInputBinding
  - caution: drop of support for ViewEngine libs
- 17: New control flow syntax incl. @defer (mig.!)
  - View Transition API
- 18: Zoneless
  - SSR: Event Replay
  - @let
- 19: Hybrid Rendering & Incremental Hydration
  - SSR & @defer: hydrate on
  - LinkedSignal & Ressouce API

NG Updates

Standalone Components

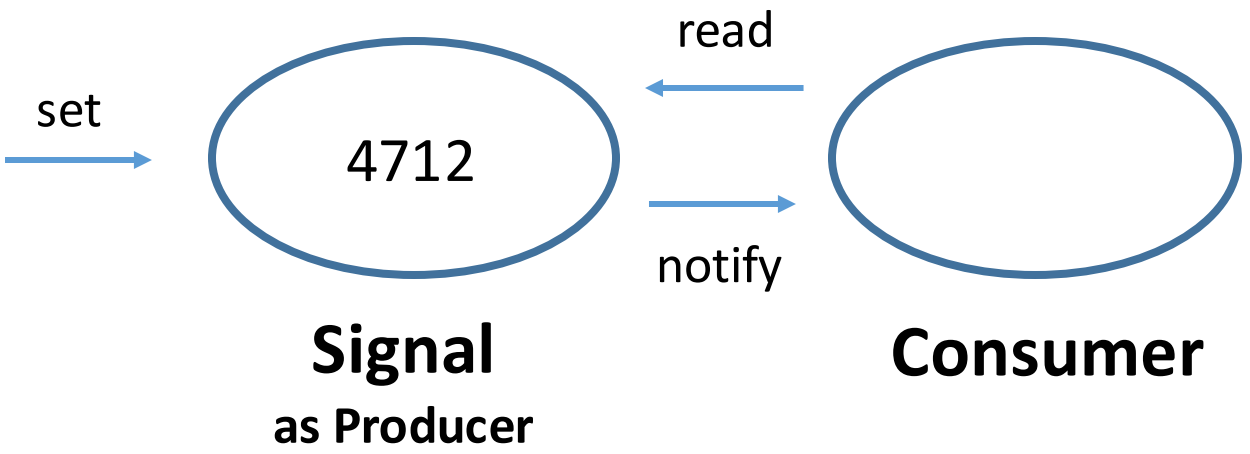
- Becoming the standard

- Good for performance because
  - fine-grained lazy-loading
  - @defer & hydrate on
  - making NgZone obsolete in the future!?

- Use migration script!

```
ng generate @angular/core:standalone
```

Signals



Signals – Component without signals

```
flights: Flight[] = [];
```

```
const flights = await this.flightService.findAsPromise(from, to);
this.flights = flights;
```

```
<div *ngFor="let f of flights">
    <flight-card [item]="f" />
</div>
```

Signals – Component using signals

```
flights = signal<Flight[]>([]);
```

```
const flights = await this.flightService.findAsPromise(from, to);
this.flights.set(flights);
```

```
<div *ngFor="let f of flights()">
    <flight-card [item]="f" />
</div>
```

Signals – RxJS Interop

toObservable(signal)

toSignal(observable$)

DestroyRef and takeUntilDestroyed() op.

```
@Component({...})
export class AppComponent {
  constructor() {
    inject(DestroyRef).onDestroy(() => {
      // Writte your cleanup logic
    })
  }
}
```

Only available in the constructor()

```
const sub = this.store.select(getUser()).pipe(takeUntilDestroyed())
  .subscribe((user) => {
  this.user = user
});
```

Non Destructive Hydration

- Server Side Rendering
- NG <= 15 the complete DOM was destroyed and rerendered
- NG => 16 the DOM is being hydrated with event handlers
- Thus we don’t have a flash → better UX

Vite + esbuild

- Much faster ng serve / ng build

- Give it a try by changing the builder in angular.json
  - NG 16:
    - "builder": "@angular-devkit/build-angular:browser-esbuild",
  - NG >= 17:
    - "builder": "@angular-devkit/build-angular:application",
    - drop off several build flags
      - vendorChunk
      - buildOptimization

Migrate to NG 17 Control Flow

- The future is here ☺
- May look a bit awkward at first sight
  - but it has (a lot) **performance** benefits and
  - on top of that it make things **easier**

- Easy migration

```
ng generate @angular/core:control-flow
```

- Make sure to add
  - @empty / @else
  - revisit all track @for

Angular 17 View Transitions

```
export const appConfig: ApplicationConfig = {
  providers: [
    provideRouter(
      routes,
      withViewTransitions() // the magic
    ),
  ]
};
```

Angular 17 View Transitions Customization

```
@keyframes fade-in {
  from { opacity: 0 }
}
@keyframes fade-out {
  to { opacity: 0 }
}
@keyframes slide-from-right {
  from { transform: translateX(30px) }
}
@keyframes slide-to-left {
  to { transform: translateX(-30px) }
}
```

```
::view-transition-old(root) {
  animation:
    90ms cubic-bezier(0.4, 0, 1, 1) both fade-out,
    300ms cubic-bezier(0.4, 0, 0.2, 1) both slide-to-left;
}

::view-transition-new(root) {
  animation:
    210ms cubic-bezier(0, 0, 0.2, 1) 90ms both fade-in,
    300ms cubic-bezier(0.4, 0, 0.2, 1) both slide-from-right;
}
```

Use latest Angular!

- Try to update to latest version
  - My recommendation: Wait for X.1.0 to X.2.0
- From v. 12 – 15 migration should be easy (and automatic)
- Caution with v. 16: ViewEngine support dropped for libraries
- Use https://update.angular.io

NG Updates

- 14: Typed Reactive Forms, NgOptimizedImage
- 15: Standalone Components (migration!)
- 16: Signals
  - takeUntilDestroyed() operator (& DestroyRef)
  - SSR: Non Destructive Hydration
  - required @Input()
  - Vite & esbuild
  - caution: drop of support for ViewEngine libs
- 17: New control flow syntax incl. @defer (mig.!)
  - View Transition API
- 18: Zoneless
  - SSR: Event Replay
  - @let
- 19: Hybrid Rendering & Incremental Hydration
  - SSR & @defer: hydrate on

ANGULAR
ARCHITECTS