ANGULAR
ARCHITECTS
INSIDE KNOWLEDGE

Angular's Future with Signals

LX_T

Agenda

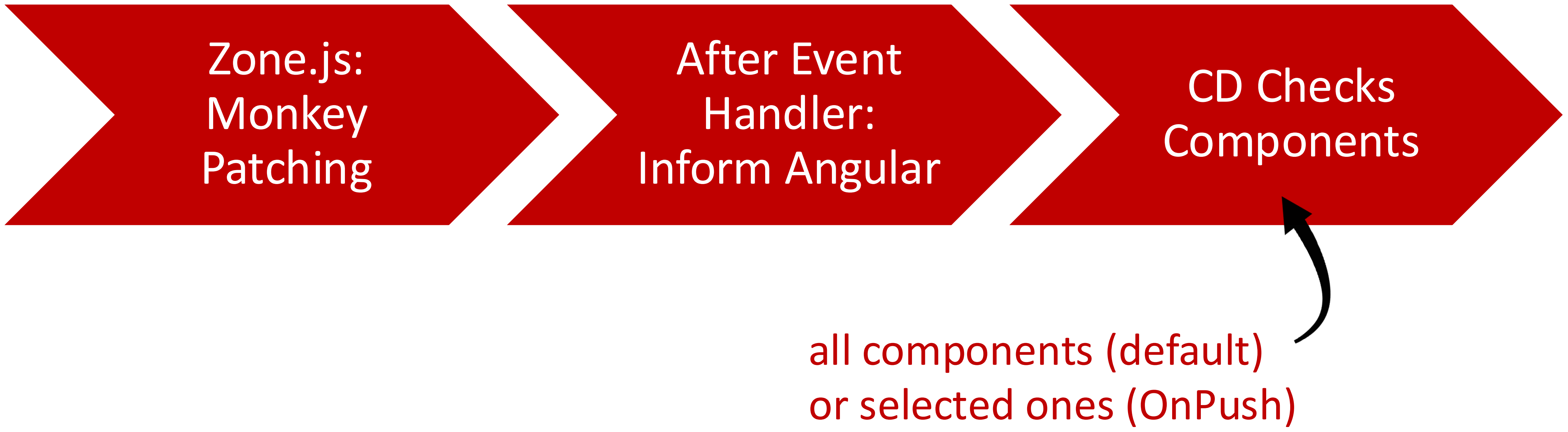
#1
Motivation
& Basics

#2
DEMO

#3
RxJS/NGRX
Interop

Motivation & Basics

Change Detection (CD) in Angular


Zone.js: Monkey Patching
After Event Handler: Inform Angular
CD Checks Components
all components (default) or selected ones (OnPush)

Drawbacks

Zone.js: Magic

Zone.js: ~100K

Cannot patch async/await

coarse grained CD

e.g. Components are always checked as a whole, even if only a tiny fraction changed

How Do Other Frameworks Solve This?

SOLID

SVELTE

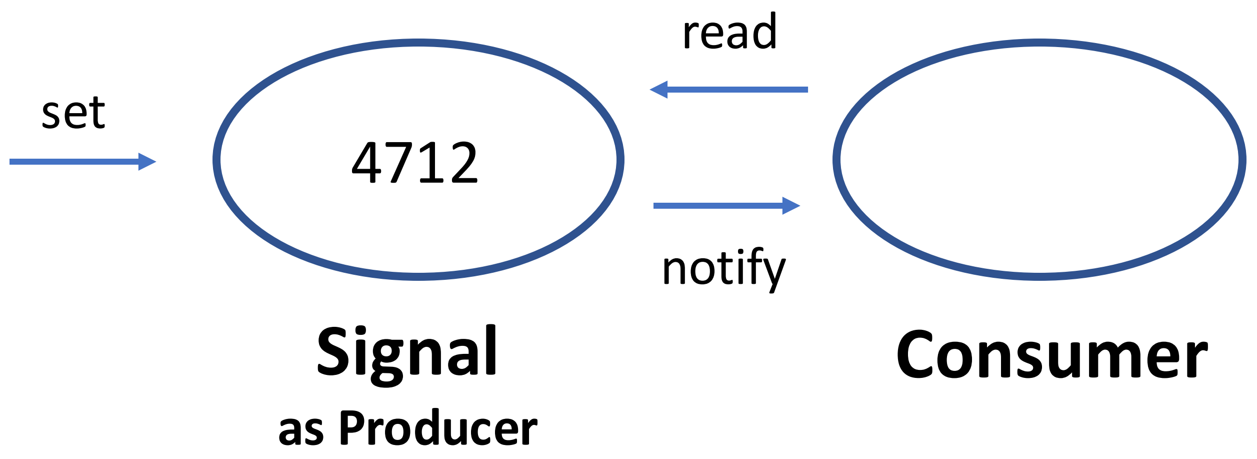
qwik

Signals!

How Will Angular Solve This?

Signals!

Signals: Simple Reactivity



Component Before Signals

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

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

```
@for (flight of flights; track flight) {
    <app-flight-card [item]="flight" />
}
```

Component With Signals

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

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

```
@for (flight of flights(); track flight) {
    <app-flight-card [item]="flight" />
}
```

Signals (field initializer, set, get & update)

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

```
this.flights.set(flights);
```

```
this.flights()
```

```
@for (flight of flights(); track flight.id) {
    <app-flight-card [item]="flight" />
}
```

```
this.flightsSignal.update((flights) => [...flights]);
```

#2:
DEMO

ANGULAR ARCHITECTS
INSIDE KNOWLEDGE

Signals (computed & effect)

```
readonly flightsLength = computed(() => this.flightsSignal().length);
```

```
effect(() => console.log(this.flightsLength() + ' flight(s) found.'));
```

#2:
DEMO

ANGULAR ARCHITECTS
INSIDE KNOWLEDGE

Subjects vs Signals – Details

RxJS Subjects (Eventing, State, Comparing)	**Angular Signals (State)**
Complex usage	Lightweight usage (especially getting current value)
Subscription management necessary	No subscription needed (done internally)
More features	Less powerful
Choose between • Eventing/Messaging (Subject) • State (BehaviorSubject) or • Comparing (ReplaySubject)	No choices, clearly opinionated
RxJS provides a **ton of functionality** to operate on observables like the map, filter, debounceTime & distinctUntilChanged, delay and retry operators	**Angular** provides two operators: • effect() *like tap() and subscribe* and • computed() *for all others* ☺
Using multiple subjects may lead to glitches	Diamond problem solved using multiple signals
RxJS currently by HTTP Client, Forms & Router	Optional for component inputs, outputs and queries

RxJS and Signal Interoperability

ANGULAR ARCHITECTS
INSIDE KNOWLEDGE

@angular/core/rxjs-interop

- toObservable(signal)
- toSignal(observable$)
- takeUntilDestroyed()
- outputFromObservable()

DEMO

ANGULAR ARCHITECTS
INSIDE KNOWLEDGE

Conclusion

Fine-grained CD

Zone-less Future

Convertible to Observables and vice versa!

No need to unsubscribe!

No need to update code!