Angular Under the Hood

Marko Stanimirović

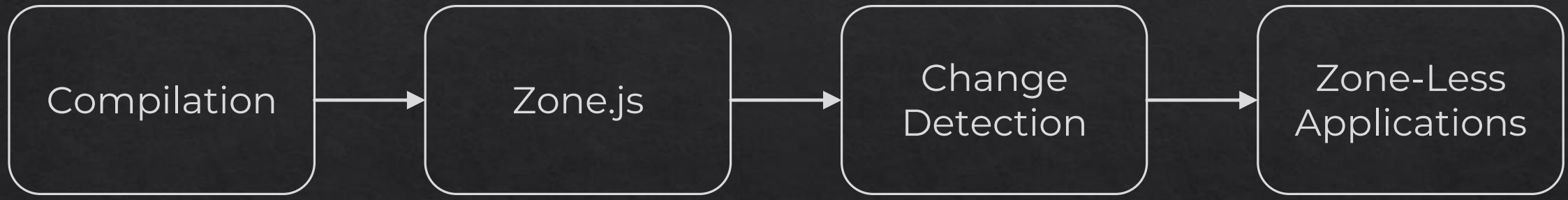
Marko Stanimirović

- Software Developer
- Angular Belgrade Organizer
- OSS Contributor
- Hobby Musician
- MSc in Software Engineering

@MarkoStDev

markostanimirovic

Contents


Compilation
Zone.js
Change Detection
Zone-Less Applications

Compilation

```ts
import { Component } from '@angular/core';

@Component({
  selector: 'app-home',
  template: `
    <h1>Home</h1>
    <h2>{{ subtitle }}</h2>
  `,
})
export class HomeComponent {
  subtitle = 'Hello World!';
}
```

ng build

```ts
class HomeComponent {
  constructor() {
    this.subtitle = 'Hello World!';
  }
}

HomeComponent.ɵfac = function HomeComponent_Factory(t) {
  return new (t || HomeComponent);
};
HomeComponent.ɵcmp = ɵɵdefineComponent({
  type: HomeComponent,
  selectors: [['app-home']],
  template: function HomeComponent_Template(rf, ctx) {
    if (rf & 1) { // ɵRenderFlags.Create
      ɵɵelementStart(0, 'h1'); // <h1>
      ɵɵtext(1, 'Home'); // Home
      ɵɵelementEnd(); // </h1>
      ɵɵelementStart(2, 'h2'); // <h2>
      ɵɵtext(3); // {{ subtitle }}
      ɵɵelementEnd(); // </h2>
    }
    if (rf & 2) { // ɵRenderFlags.Update
      ɵɵadvance(3); // advance three times to '{{ subtitle }}' node
      ɵɵtextInterpolate(ctx.subtitle); // update '{{ subtitle }}' node
    }
  },
});
```

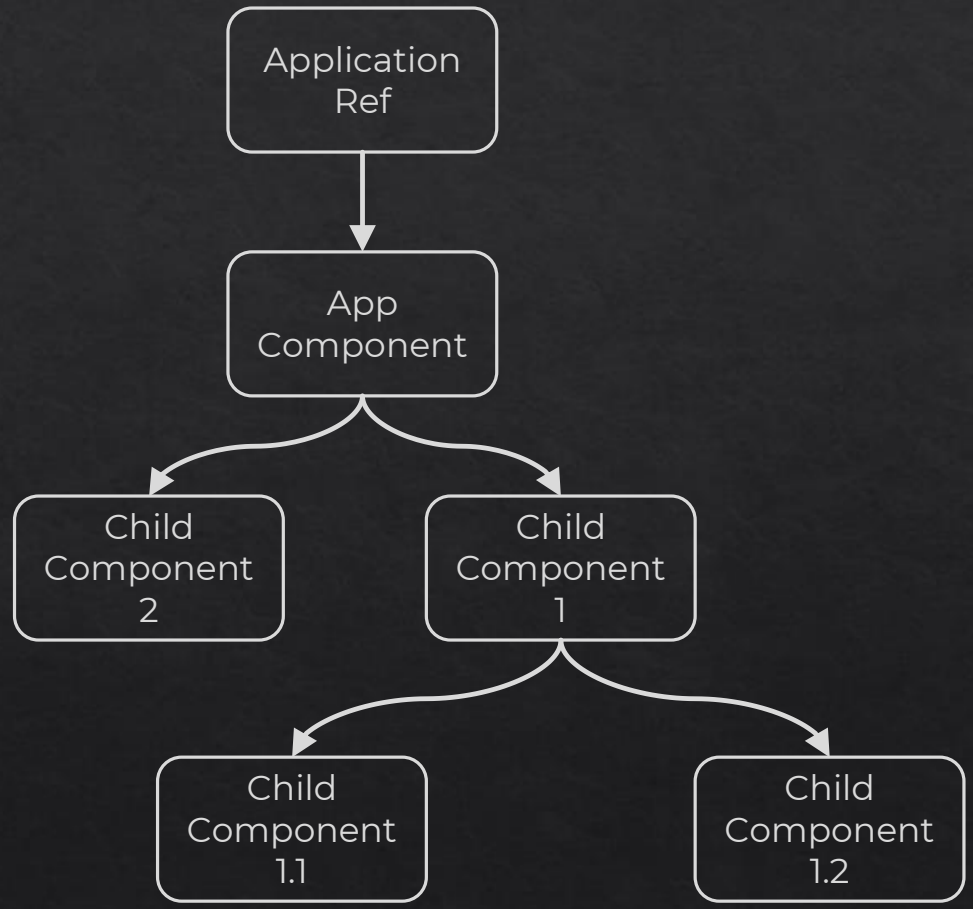
Zone.js

- Executed before the Angular application is bootstrapped
- Tells Angular when to run the change detection mechanism
- Monkey patches browser APIs

```
const originalSetTimeout = setTimeout;

setTimeout = (callback, ms) => {
  const patchedCallback = () => {
    callback();
    console.log('Tell Angular to run change detection!');
  };

  originalSetTimeout(patchedCallback, ms);
};

// usage
setTimeout(() => console.log('Timeout ended!'), 1000);

// console:
// Timeout ended!
// Tell Angular to run change detection!
```

Change Detection

```
@Injectable()
export class ApplicationRef {
  constructor(private _zone: NgZone) {
    this._zone.onMicrotaskEmpty.subscribe(() =>
      this._zone.run(() => this.tick());
    );
  }
}
```



Zone.js Downsides

- Not tree shakable (uncompressed size >100kB)
- Slower application bootstrap speed
- Unnecessary change detection triggering
- Magic
- Cannot monkey patch native async/await (ES2017)

Zone-Less Applications

- Trigger change detection only when necessary

```
// polyfills.ts
// import 'zone.js/dist/zone'; 👈

// main.ts
platformBrowserDynamic()
  .bootstrapModule(AppModule, { ngZone: 'noop' }) // 👈
  .catch(err => console.error(err));
```

```
import { Component, ɵmarkDirty } from '@angular/core';

@Component({
  selector: 'app-counter',
  template: `
    <button (click)="onIncrement()">+</button>
    <span>{{ count }}</span>
    <button (click)="onDecrement()">–</button>
  `,
})
export class CounterComponent {
  count = 0;

  onIncrement(): void {
    this.count++;
    ɵmarkDirty(this); // 👈
  }

  onDecrement(): void {
    this.count--;
    ɵmarkDirty(this); // 👈
  }
}
```

How to trigger change detection automatically?

```ts
import { Component } from '@angular/core';
import { BehaviorSubject } from 'rxjs';

@Component({
  selector: 'app-counter',
  template: `
    <button (click)="onIncrement()">+</button>
    <span>{{ count | zoneLessAsync }}</span> <!— 👈 —>
    <button (click)="onDecrement()">—</button>
  `,
})
export class CounterComponent {
  readonly count = new BehaviorSubject(0);

  onIncrement(): void {
    this.count.next(this.count.value + 1);
  }

  onDecrement(): void {
    this.count.next(this.count.value - 1);
  }
}
```

Demo

Thank you!