INFINUM

Angular 2

MIHAEL ŠAFARIĆ

INFINUM

~~Angular 2~~

Angular 4

MIHAEL ŠAFARIĆ

INFINUM

~~Angular 2~~

~~Angular 4~~

Angular

MIHAEL ŠAFARIĆ

ANGULAR VS ANGULARJS

- not backwards compatible
- new era of frontend development
- ngUpgrade

WHY ANGULAR?

- easy to learn
- performance
- component based web development

ANGULAR FEATURES

Cross Platform

- Progressive web apps
- Desktop
- Native

Speed and Performance

- Code generation
- Code splitting
- Server-side rendering

Productivity

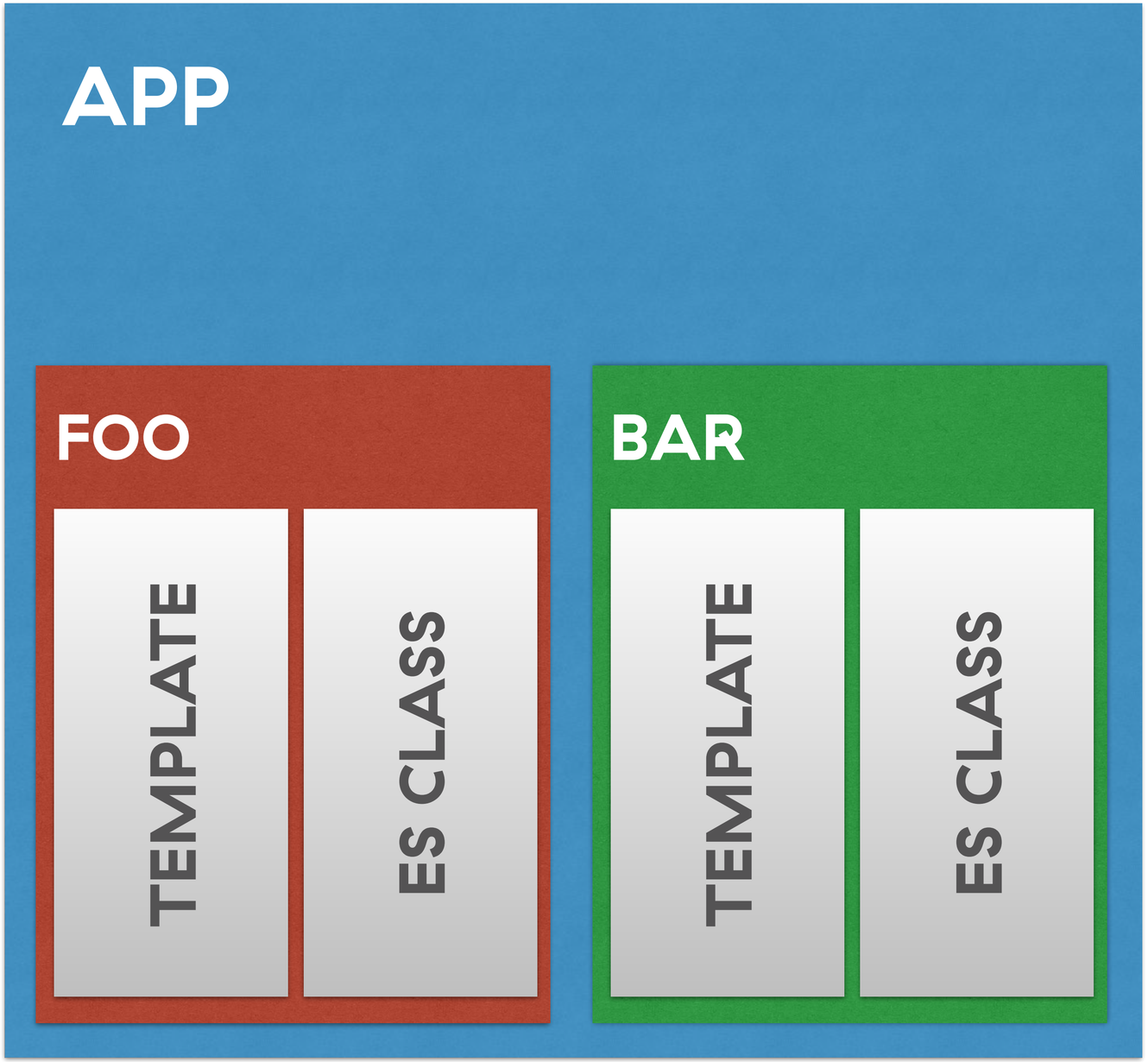
- Typescript
- Angular CLI

01

APPLICATION STRUCTURE

BUILDING BLOCKS OF AN ANGULAR APP

- Modules
- Components
- Templates
- Metadata
- Data binding
- Directives
- Services
- Dependency injection



APP

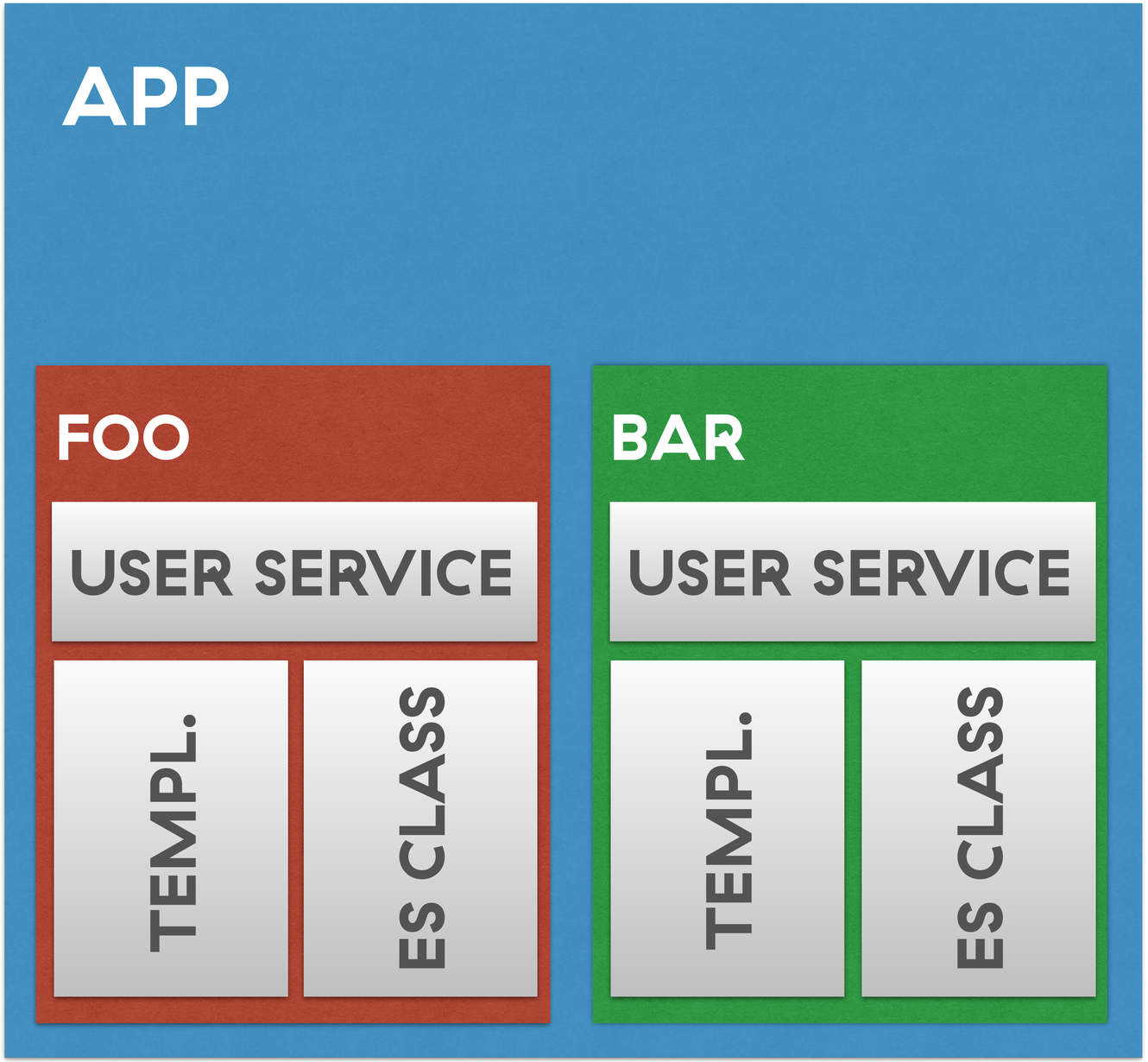
COMPONENTS

FOO

- TEMPLATE
- CLASS

BAR

- TEMPLATE
- CLASS



APP

COMPONENTS

FOO

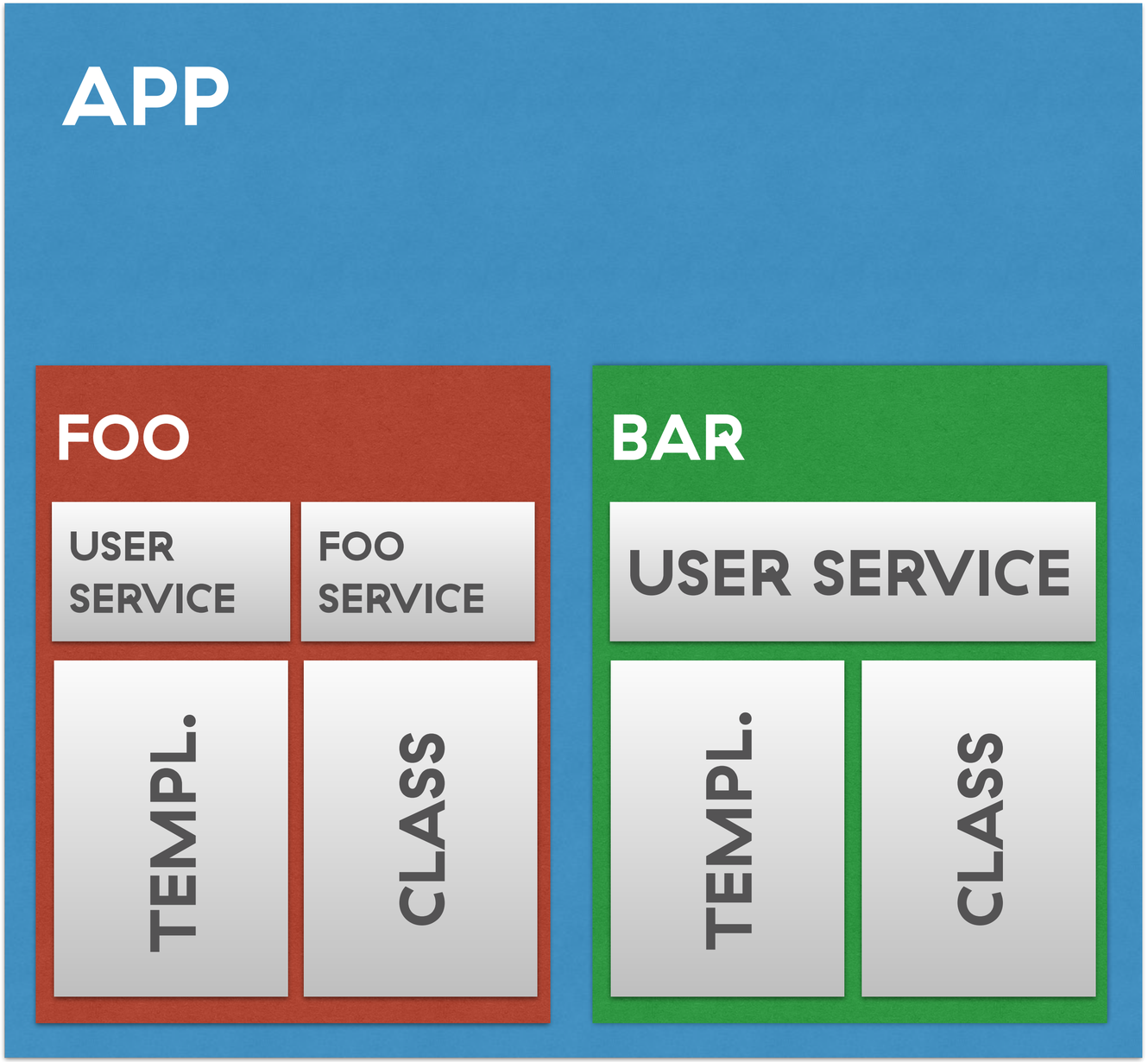
- TEMPLATE
- CLASS

BAR

- TEMPLATE
- CLASS

SERVICES

USER.SERVICE



APP

COMPONENTS

FOO

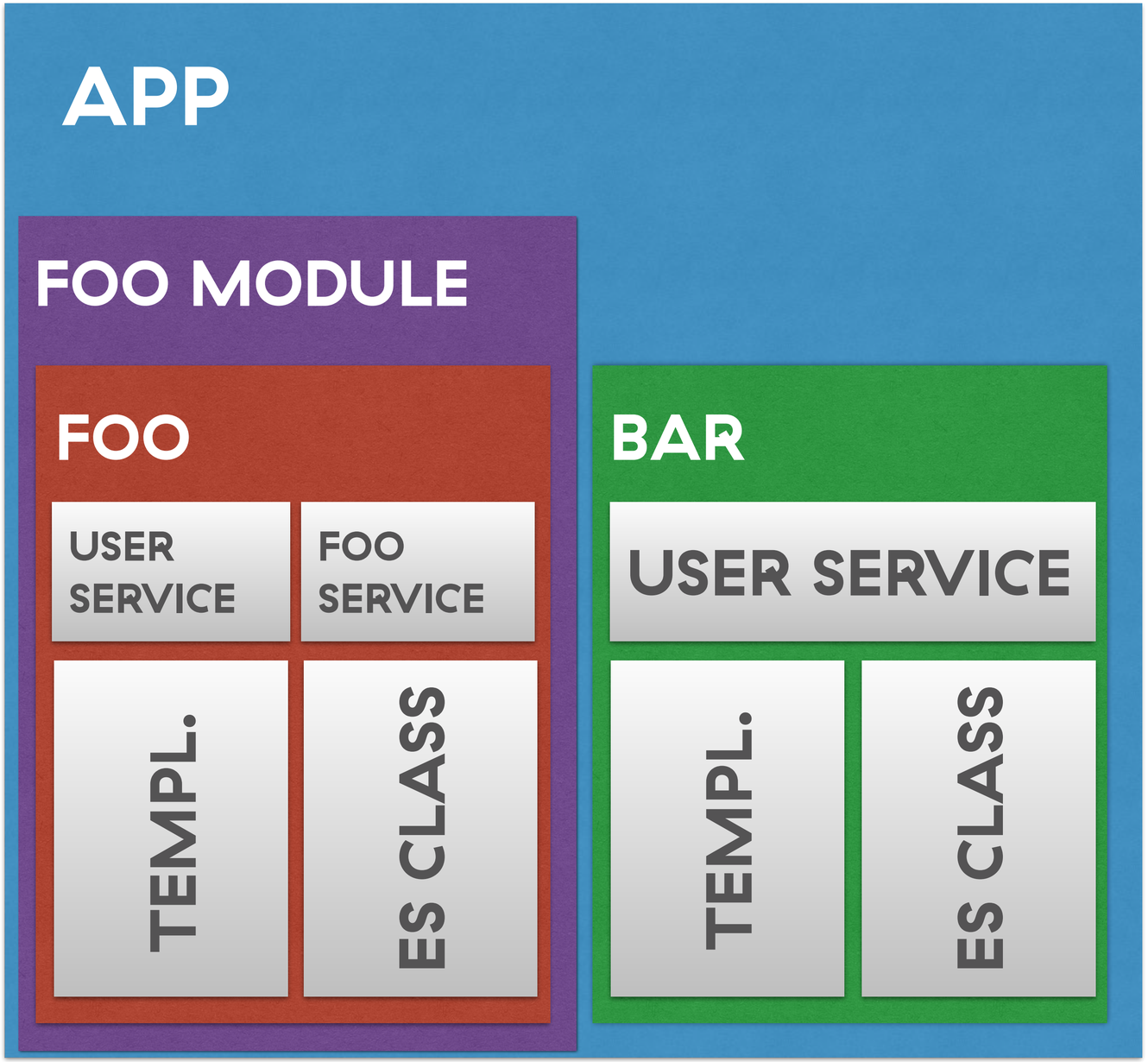
- TEMPLATE
- CLASS
- FOO.SERVICE

BAR

- TEMPLATE
- CLASS

SERVICES

- USER.SERVICE



APP

COMPONENTS

FOO

- TEMPLATE
- CLASS
- FOO.SERVICE
- FOO.MODULE

BAR

- TEMPLATE
- CLASS

SERVICES

USER.SERVICE

02

COMPONENTS

STRUCTURE

- Reusable UI building block
- Component tree
- Template + component class

COMPONENT CLASS

```
export class DuckyComponent {

}
```

COMPONENT CLASS

```
@Component({


})
export class DuckyComponent {

}
```

COMPONENT CLASS

```
@Component({
  selector: 'ducky-component',
  template: '<p>Hello {{name}}</p>'
})
export class DuckyComponent {
  name: string = 'Awesome duck';
}
```

USING A COMPONENT

```
@Component({
  selector: 'ducky-component',
  template: '<p>Hello {{name}}</p>'
})
export class DuckyComponent {
  name: string = 'Awesome duck';
}
```

```
<ducky-component></ducky-component>
```

PASSING DATA INTO COMPONENT

```
@Component({
  selector: 'ducky-component',
  template: '<p>Hello {{name}}</p>'
})
export class DuckyComponent {
  @Input() name: string;
}
```

```
<ducky-component name="Awesome duck"></ducky-component>
```

EVENT CATCHING

```
@Component({
  selector: 'ducky-component',
  template: '<p (click)=“increment()">Ducks: {{count}}</p>'
})
export class DuckyComponent {
  count: number = 0;

  increment() {
    this.count++;
  }
}
```

EVENT PROPAGATION

```
@Component({
  selector: 'ducky-component',
  template: '<button (click)="incrementDucks()"></button>'
})
export class DuckyComponent {
  @Output() result: EventEmitter<number> = new EventEmitter();

  incrementDucks() {
    this.result.emit(this.count);
  }
}
```

EVENT PROPAGATION

```
<ducky-component (result)="doSomethingWithDucks($event)">
</ducky-component>
```

STRUCTURAL DIRECTIVES

```
<div *ngIf="duck.isAlive">Quack!</div>
```

```
<div *ngFor="let duck of ducks">{{ duck.name }}</div>
```

ATTRIBUTE DIRECTIVES

```
<div bgColor="blue">I'm blue</div>
```

```
@Directive({ selector: 'bgColor' })
export class BackgroundImageDirective {
  constructor(private el: ElementRef) { }

  @Input()
  set bgColor(color: string) {
    this.el.nativeElement.style.backgroundColor = color;
  }
}
```

03

ROUTING

ROUTER

- singleton
- navigation between states
- nested routes
- auxiliary routes

04

FORMS

TWO FORM-BUILDING TECHNOLOGIES

- template-driven forms
- reactive (model driven) forms

TEMPLATE-DRIVEN FORMS

- form structure defined in template
- form specific directives (ngModel, ngForm, ngSubmit, ...)

MAPPING DUCK INSTANCE TO THE FORM

- ngModel
- name property is required

```
<form>
  <input [(ngModel)]="duck.name" name="duckName">
</form>
```

REACTIVE FORMS

- form manipulation from component class
- observe changes in form state
- data model immutability

DUCK FORM

```
const duckForm = new FormGroup({
  name: new FormControl('Awesome duck', Validator.required),
  size: new FormControl('5'),
  children: FormArray([
    new FormControl('Mini duck'),
    new FormControl('Middle duck')
  ])
});
```

MAPPING DUCK FORM TO THE DOM

```
<form [formGroup]="duckForm">
  Name: <input formControlName="name">
  Size: <input formControlName="size">

  <div *ngFor="let duck of duckForm.children">
    {{ duckControl.name }}
  </div>
</form>
```

05

SUMMARY

SUMMARY

- Google
- cutting edge technologies
- early adoption
  - external libraries

INFINUM

Thank you!

MIHAEL.SAFARIC@INFINUM.CO
@SAFO6M

Visit infinum.co or find us on social networks:

infinum.co infinumco infinumco infinum