TEST DRIVEN DEVELOPMENT (TDD)

CODING

UNIT TEST

TDD

INSTALL PROGRAM

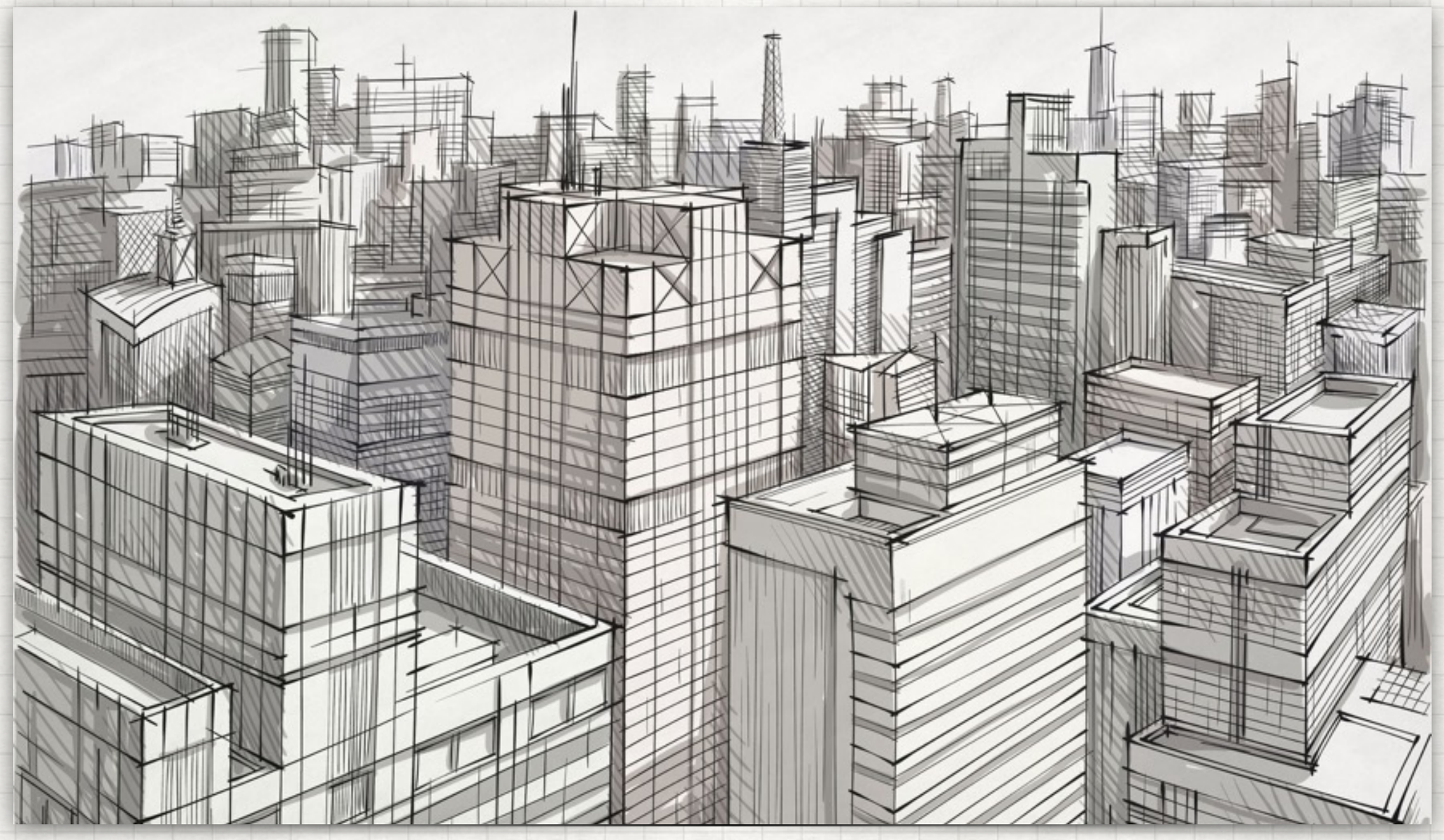
- NodeJS package manager (NPM)
  https://nodejs.org/en/download/

- TypeScript Compiler
  npm install -g typescript

- Sublime Text Plugin
  https://github.com/Microsoft/TypeScript-Sublime-Plugin

- Download Angular2 Seed
  https://github.com/angular/quickstart

TYPESCRIPT OVERVIEW



TYPESCRIPT

CLASS

```
class Greeter {

    greeting: string;

    constructor(message: string, public wow: string) { // ***public

        this.greeting = message;

    }

    greet() {

        return "Hello, " + this.greeting;

    }

}
```

TYPESCRIPT

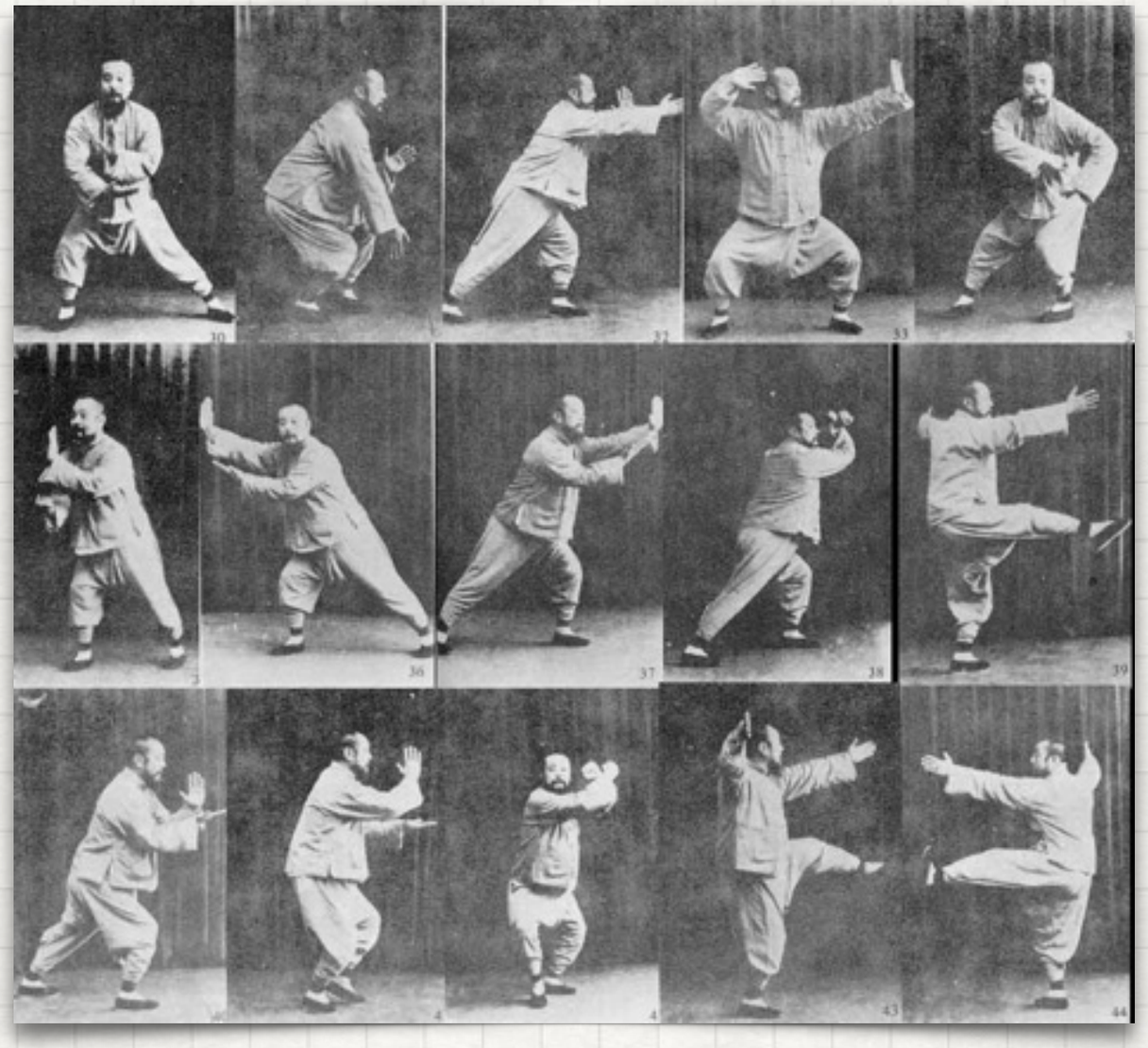
INHERITANCE

```
class Animal {
    constructor(public name: string) { }
    move(distanceInMeters: number = 0) {
        console.log(`${this.name} moved ${distanceInMeters}m.`);
    }
}
class Snake extends Animal {
    constructor(name: string) { super(name); }
    move(distanceInMeters = 5) {
        console.log("Slithering...");
        super.move(distanceInMeters);
    }
}
```

TYPESCRIPT

MORE

http://www.typescriptlang.org/



ANGULAR JS 2

WORK SHOP BY KENNY

2.0 NOW IN BETA!

OVERVIEW

- Angular2 Architecture Overview
- Component
- Life cycle
- Injectable
- How to call Data Service Api
- Event
- By pass event

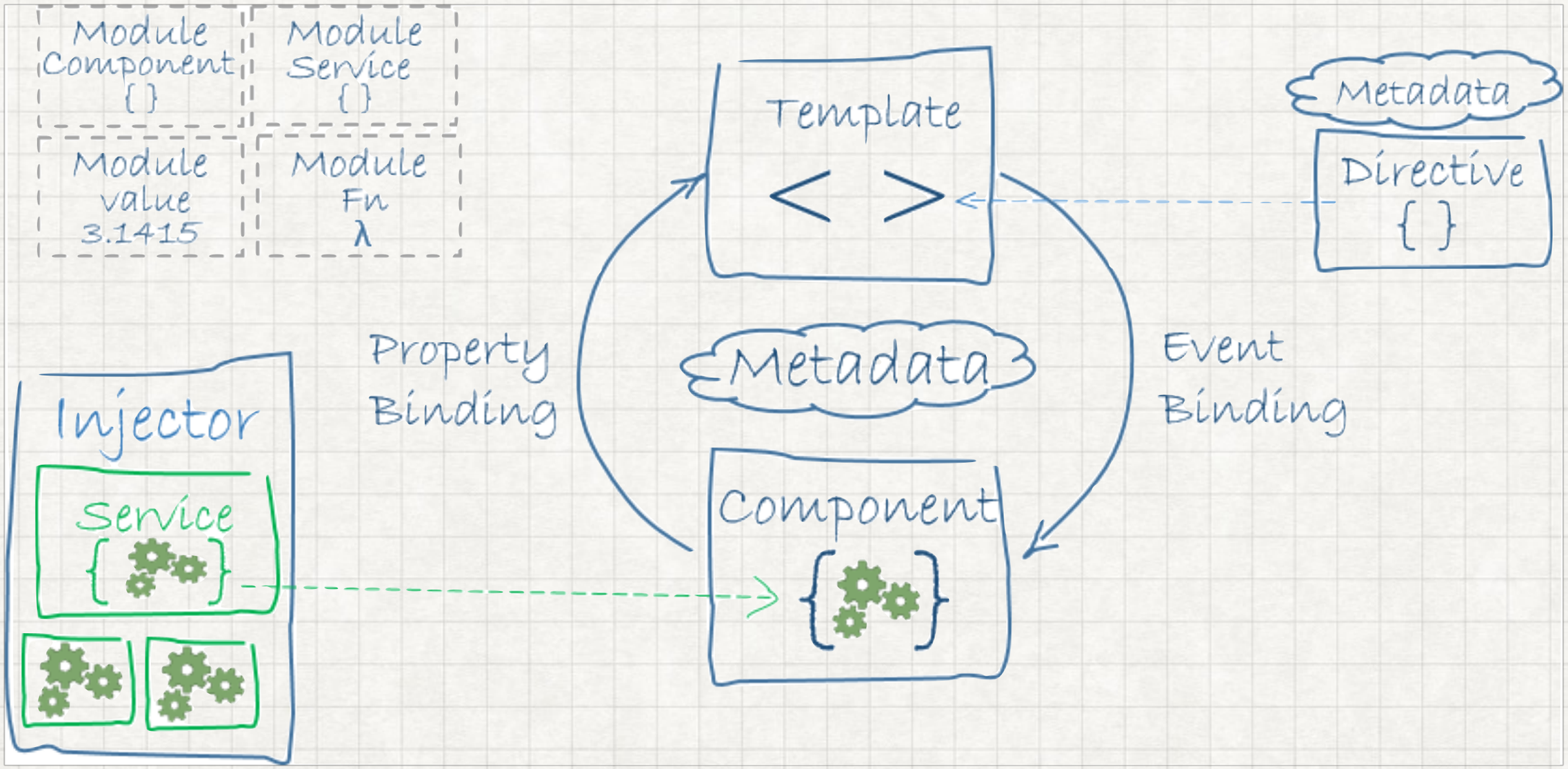
“

RUN

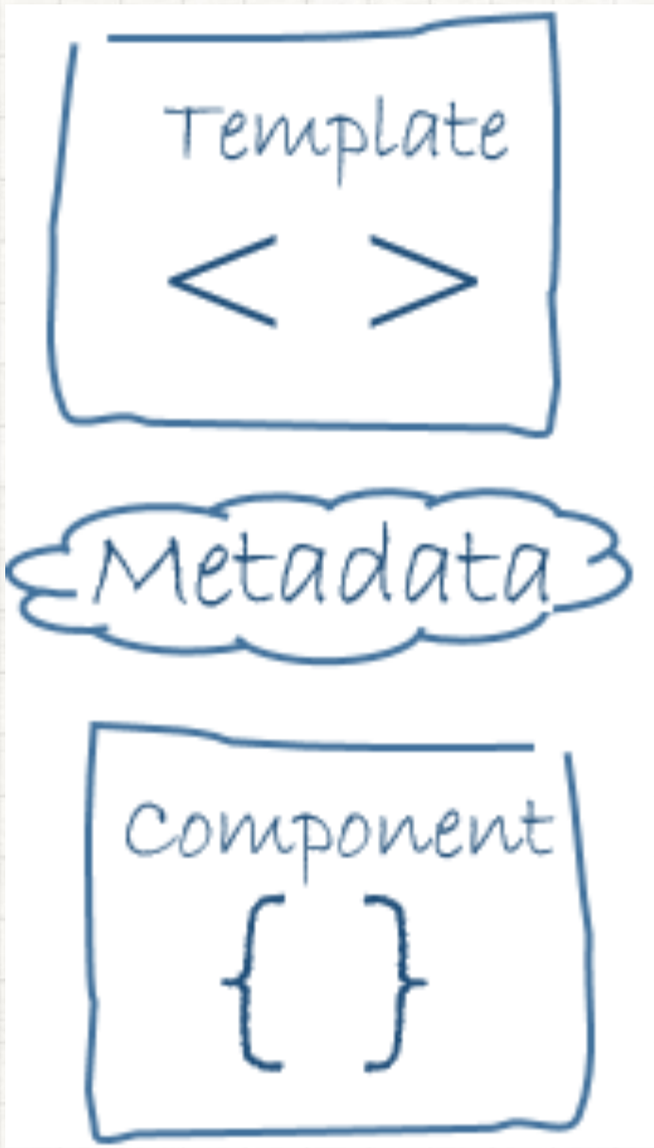
— *Monojobza*

”

ARCHITECTURE OVERVIEW



COMPONENT



“

TEMPLATE => DOM

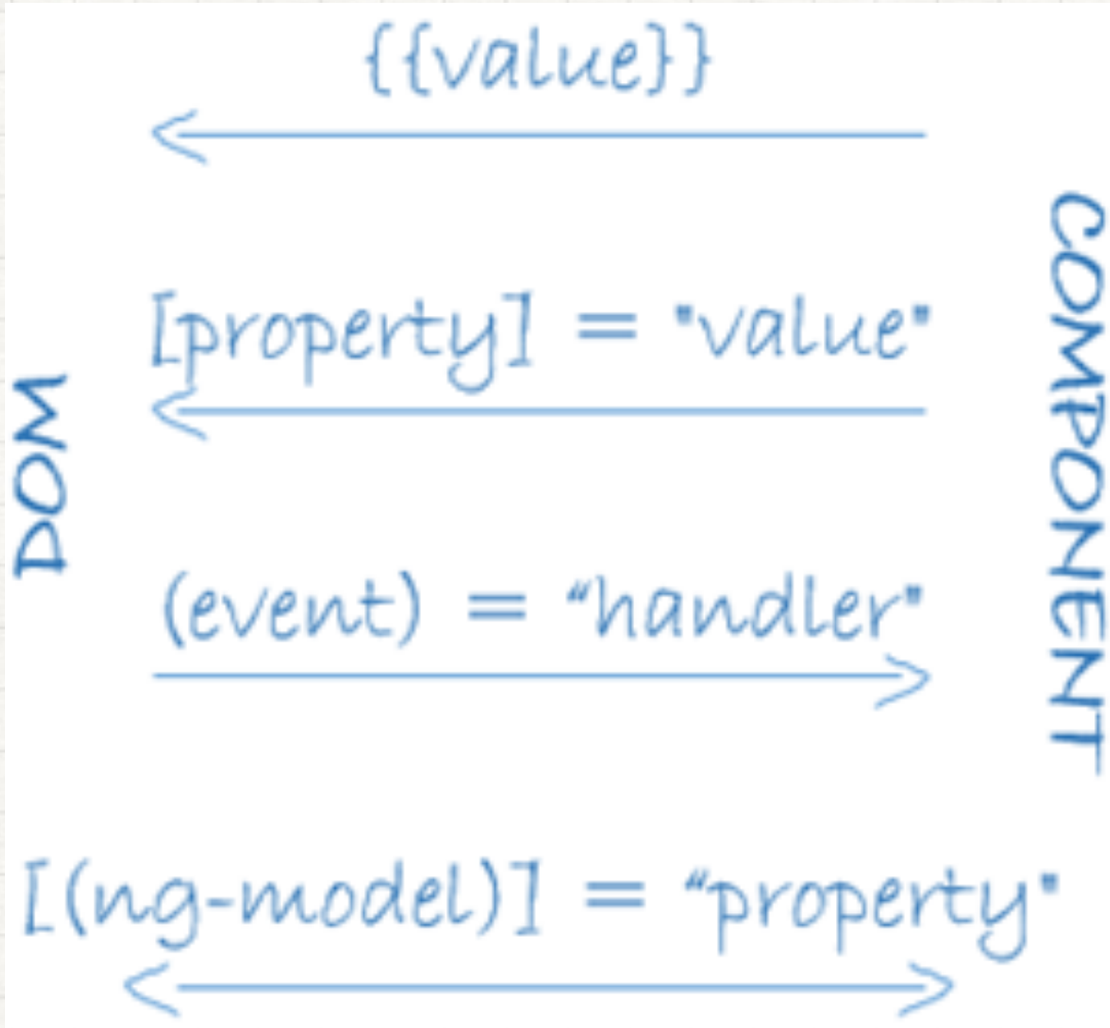
— Step D

”

```
@Component({
  selector:    'hero-list',
  templateUrl: 'app/hero-list.component.html',
  directives:  [HeroDetailComponent],
  providers:   [HeroService]
})
```

COMPONENT

CONTROLL DOM & EVENT



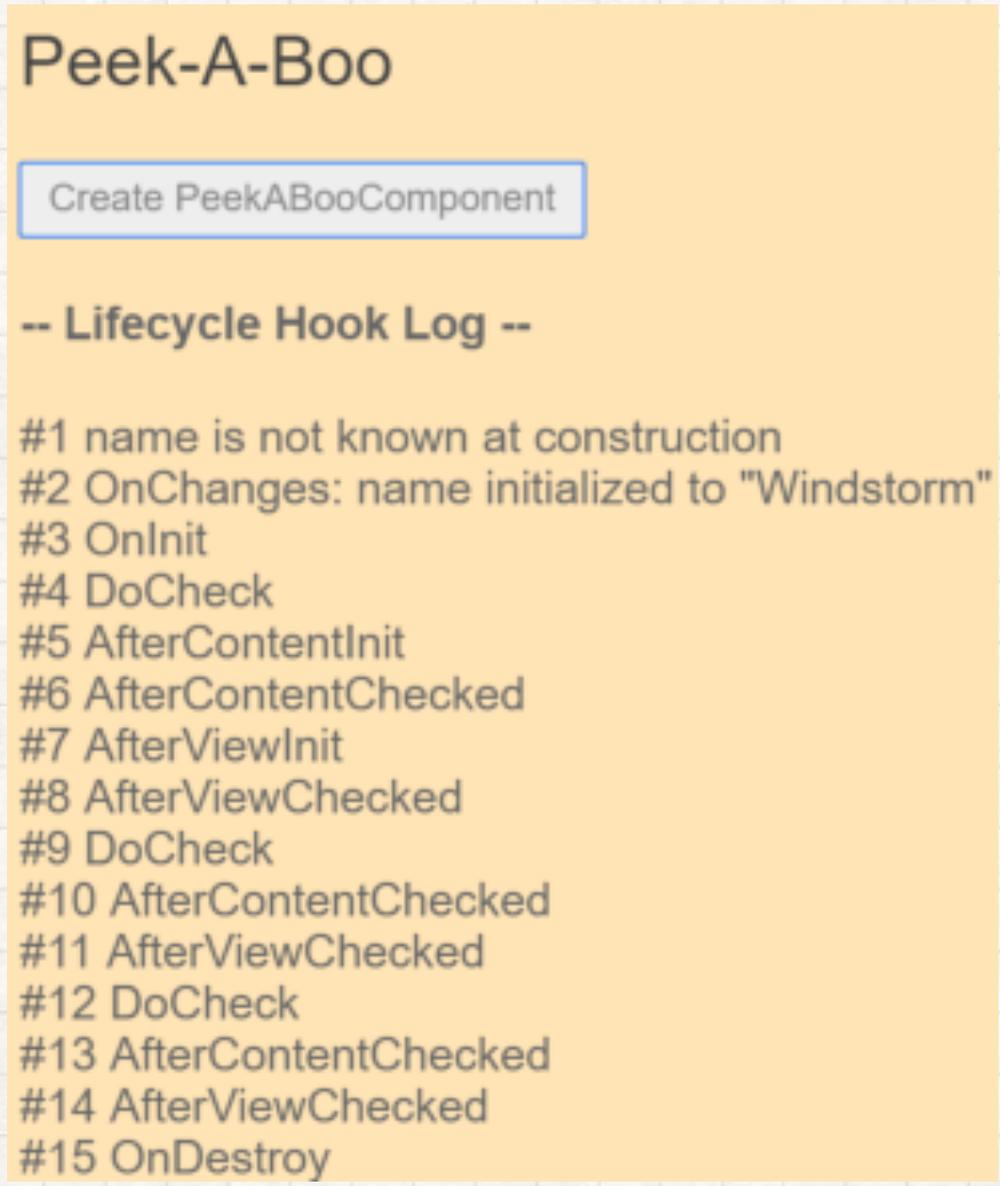
EXAMPLE

```
<div>{{hero.name}}</div>
```

```
<hero-detail [hero]="selectedHero"></hero-detail>
```

```
<div (click)="selectHero(hero)"></div>
```

COMPONENT LIFE CYCLE



“

Document.ready = ngAfterViewInit

— ไบค์เกอร์ ผู้กลัวเมีย

”

INJECTABLE

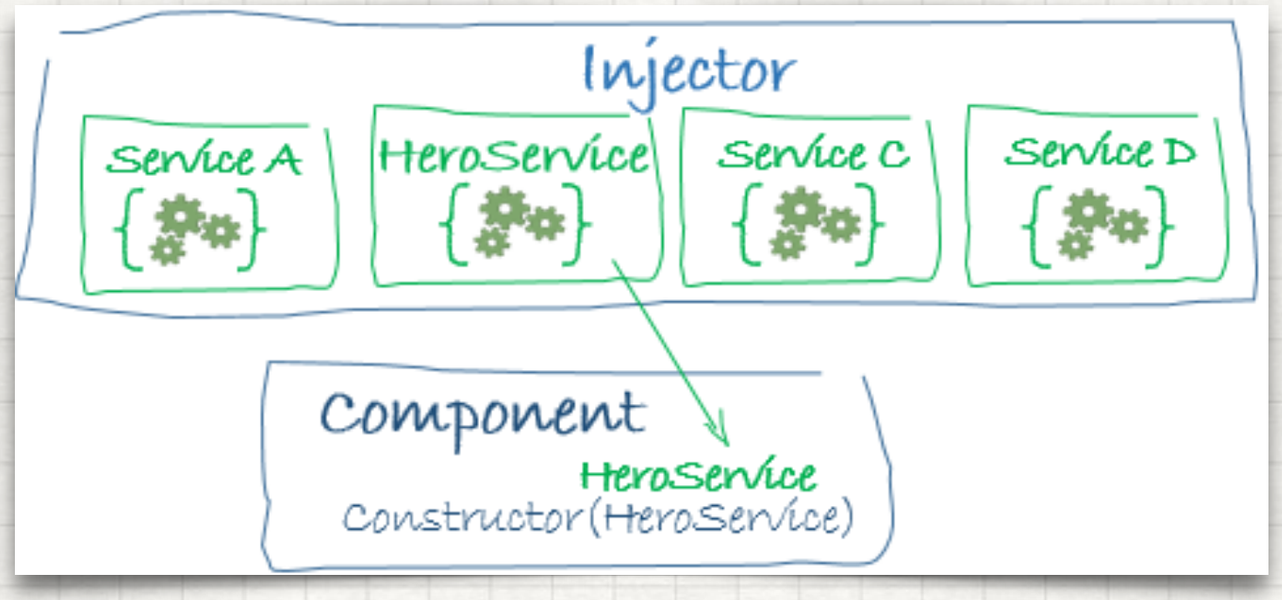
“

"Dependency Injection" is a way to supply a new instance of a class with the fully-formed dependencies it requires.

— *Teeranai Buddee*

”

INJECT ON COMPONENT CONSTRUCTOR



“

DON’T FORGET REGISTER INJECTABLE IN COMPONENT

— *P’Keaw*

”

```
@Component({

  providers:   [HeroService]

})

export class HeroesComponent {

      constructor(private _service: HeroService){ }

}
```

“

Don’t forget add @Injectable()
annotation on Injectable class

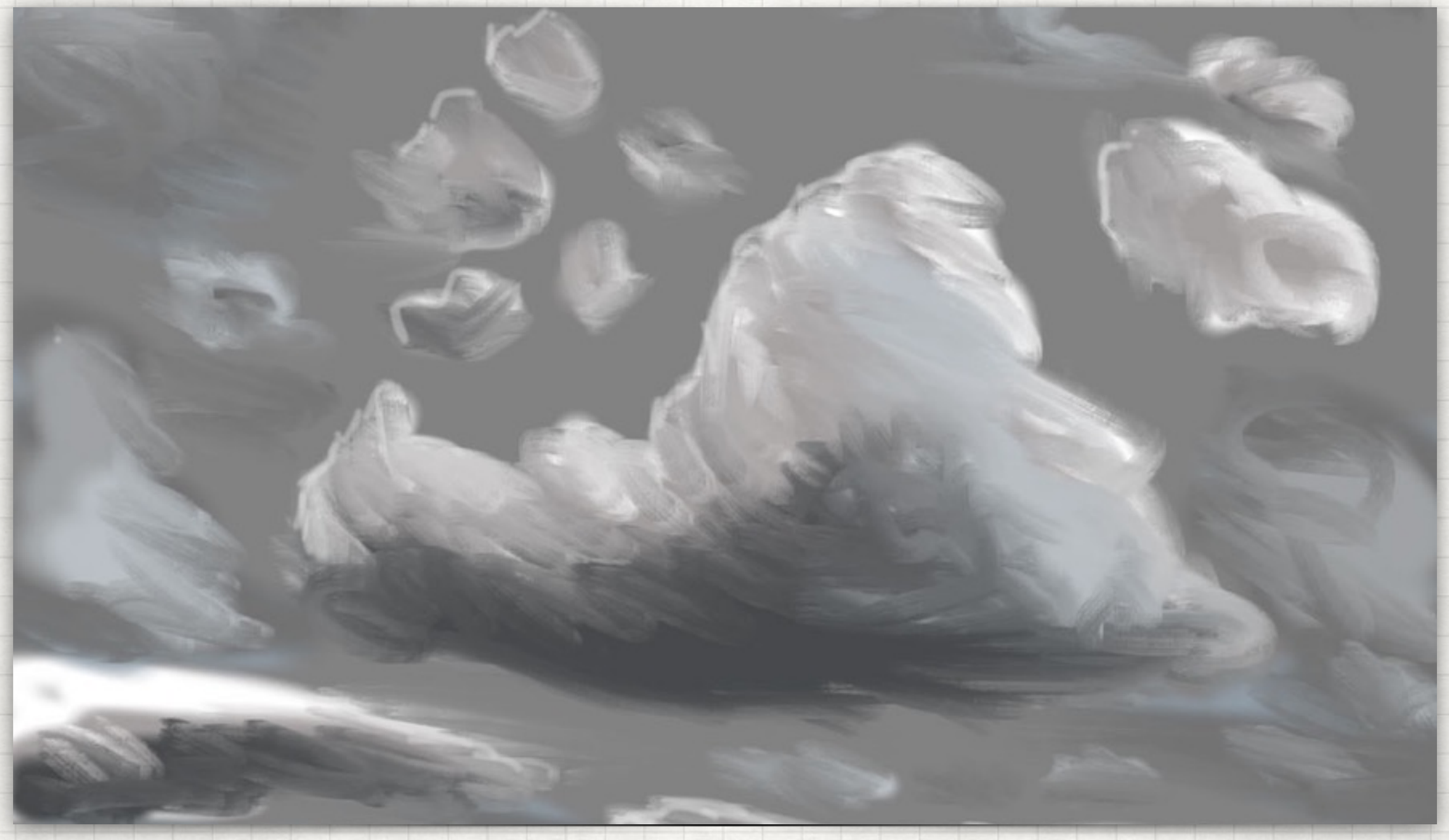
— ยมยานใหญ่

”

```
@Injectable()

export class HeroService {

  getHeroes() {

  }

}
```

HOW TO CALL DATA SERVICE API

HTTPS://ANGULAR.IO/DOCS/TS/LATEST/TUTORIAL/TOH-PT4.HTML



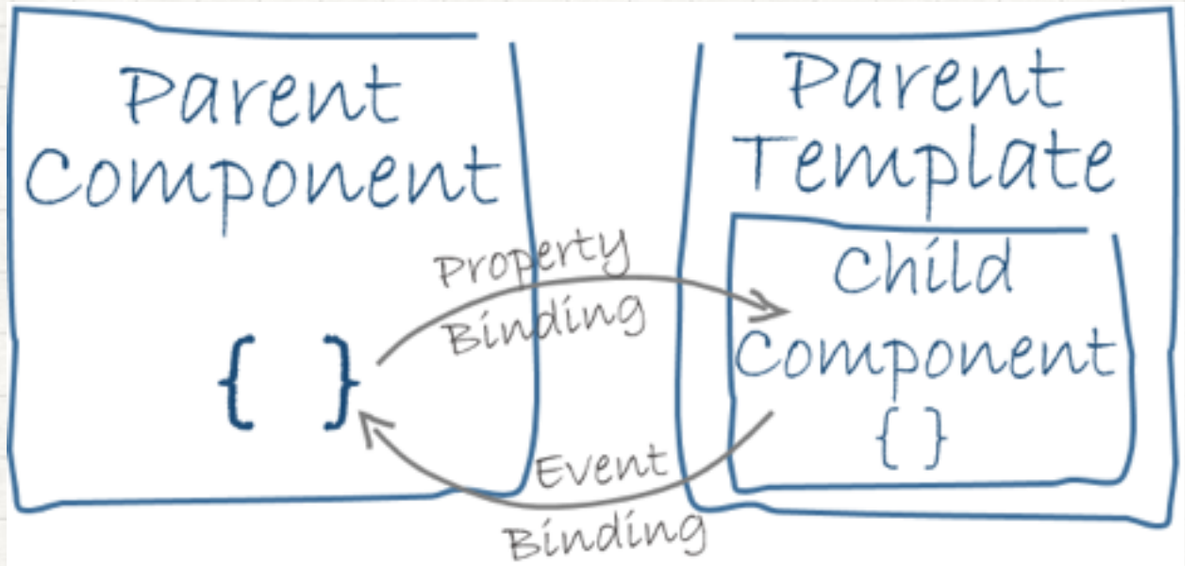
```
import {Http} from 'angular2/http';

@Injectable()
export class service{
  constructor(private http:Http) {
  }


  public GET(url:String, parameters) {
      url = this.generateGetUrlWithParameter(url, parameters)
      return new Promise(
          (resolve) => {
              this.http.get(url).subscribe(
                  (res) => {
                      resolve(res.json());
                  },
                  (err) => this.errorLogs(connectorType.GET, url, err),
                  () => this.completeLogs(connectorType.GET, url)
              )
          }
      );
  }
}
```

EVENT

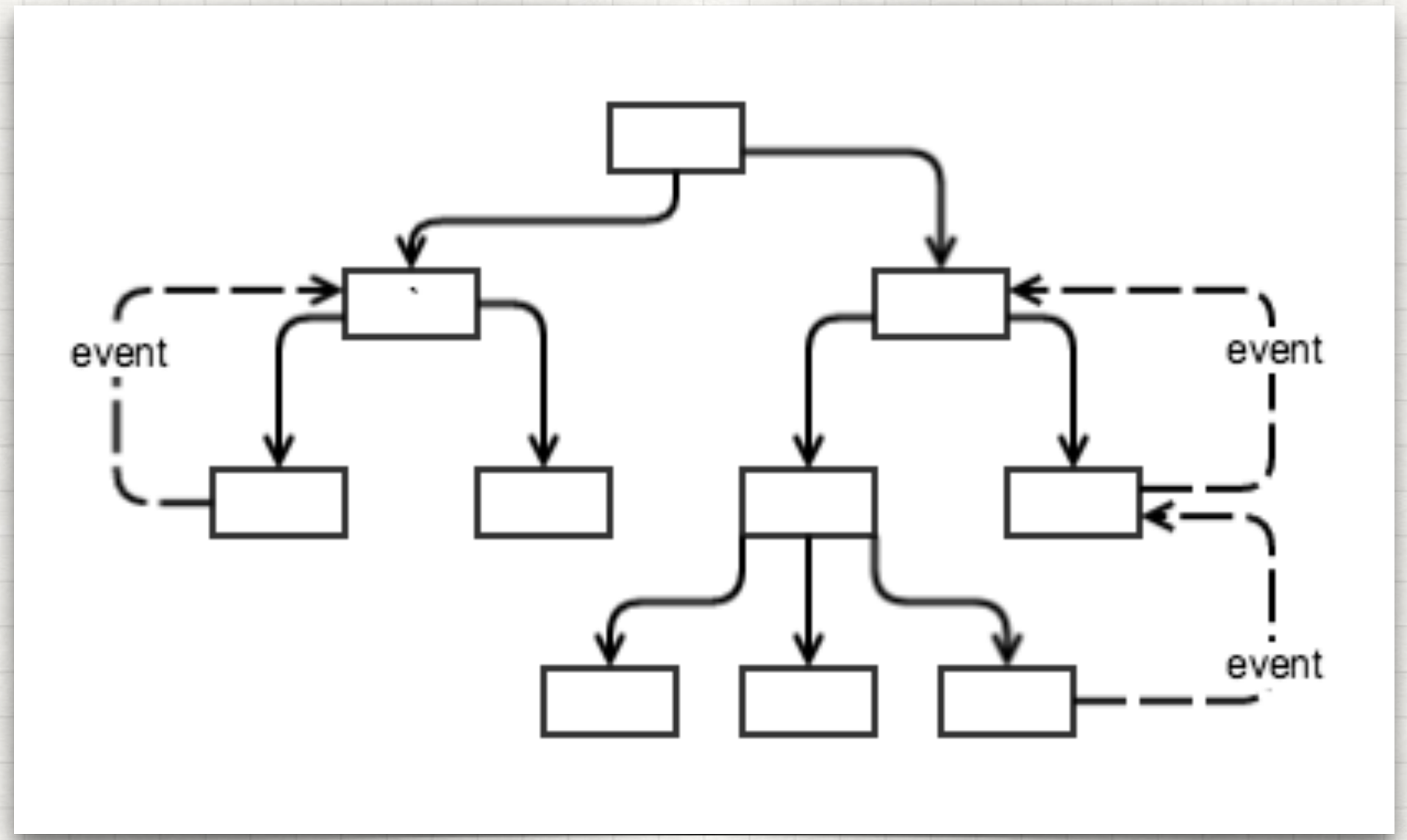
```
@Component({
    selector: 'advertise-card',
    templateUrl: `
                <div class="mdl-grid mdl-cell--12-col">
                    <div class="mdl-card__media card-image mdl-cell--4-col mdl-cell--4-col-tablet"
                         style="background: url({{card.ImageCover}}) center / cover;">
                    </div>
                    <div class="mdl-cell--8-col mdl-cell--4-col-tablet">
                        <div class="mdl-card__title">{{card.Title}}</div>
                        <div class="mdl-card__supporting-text mdl-color-text--grey-600 card-detail-preview">
                            <div></div>
                            <div>
                                <strong>{{card.HeadAds}}</strong>
                                <span>{{card.TextAds}}</span>
                            </div>
                        </div>
                            <div class="mdl-card__actions mdl-card--border card-more-detail mdl-cell--8-col mdl-cell--4-col-
                tablet">
                            <a class="mdl-button mdl-button--colored mdl-js-button mdl-js-ripple-effect button-card-right"
                (click)="viewDetail(card.Id)">
                                More Detail
                            </a>
                        </div>
                    </div>
                </div>
`,
    directives: [Dialog]
})

export class AdvertiseCard {
    item:number = 0;
    @Input() card:Card;

    constructor(private dialog:DialogService) {
    }

    private viewDetail(cardId:number) {
        var dialogProperty = new DialogProperty();
        dialogProperty.title = cardId;
        dialogProperty.content = "this is content of card Id : " + cardId;
        this.dialog.open(dialogProperty);
    }
}
```

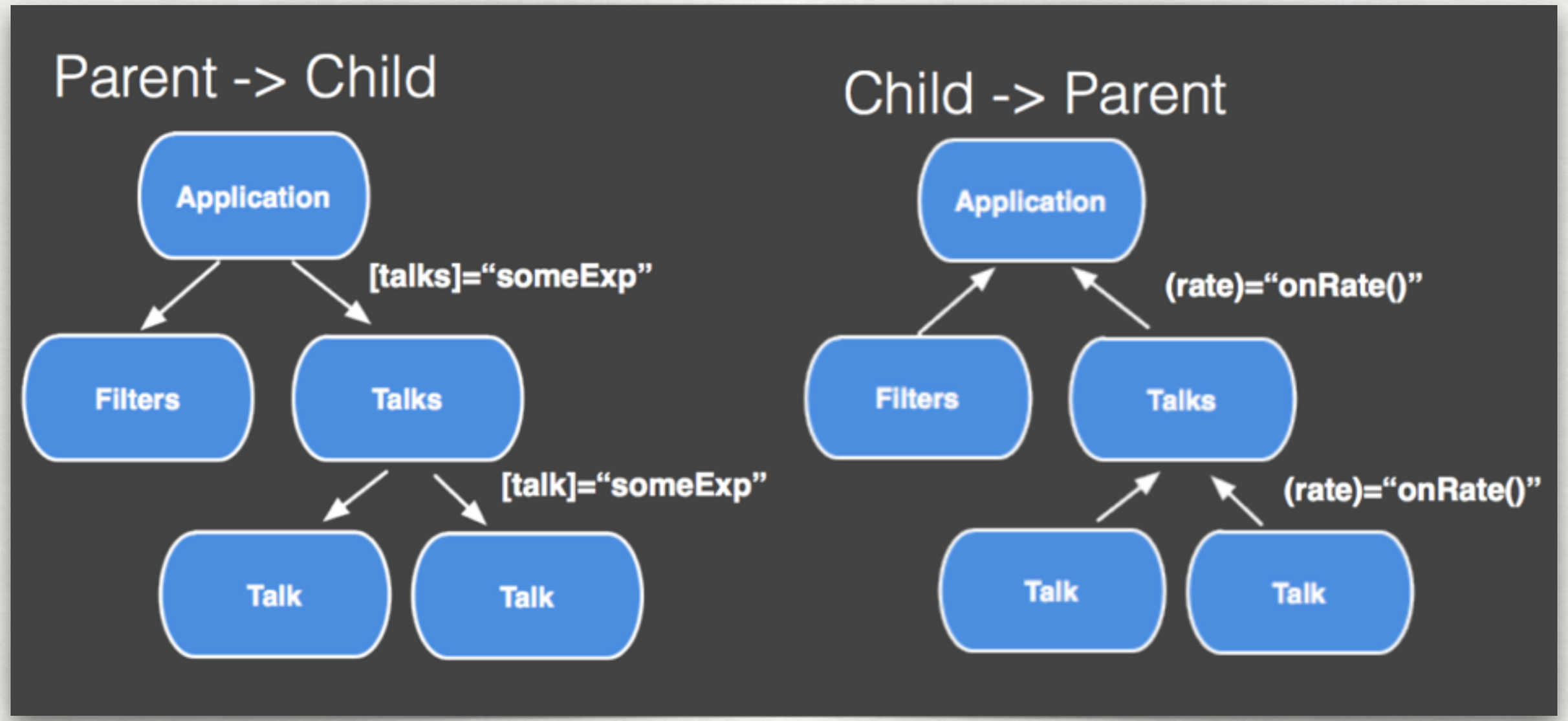
EVENT

BYPASS BETWEEN COMPONENT

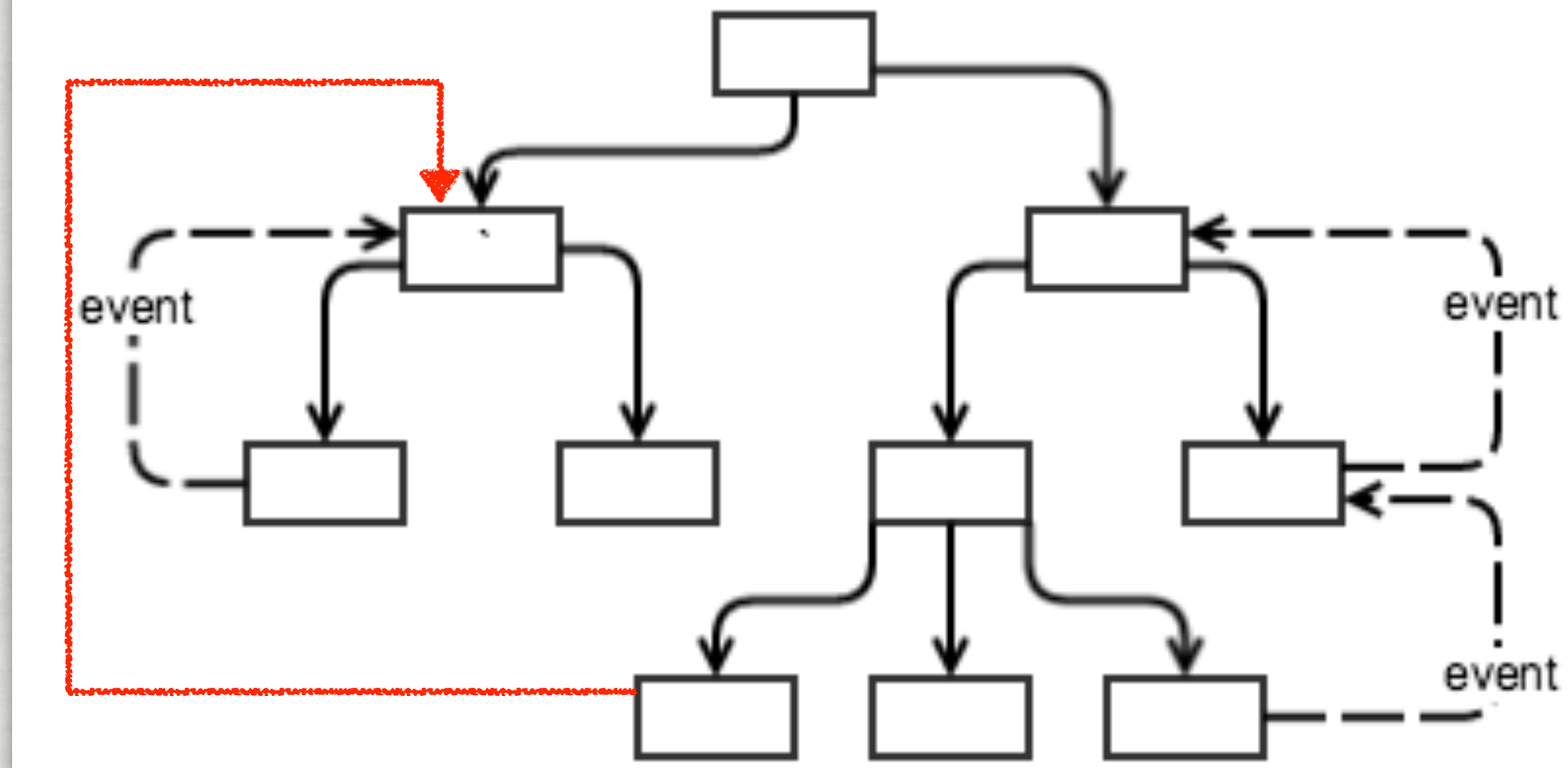


NORMAL PASS EVENT




Parent -> Child
Application
[talks]="someExp"
Filters
Talks
[talk]="someExp"
Talk
Talk
Child -> Parent
Application
(rate)="onRate()"
Filters
Talks
(rate)="onRate()"
Talk
Talk

HOW WE CALL SUPER MOTHER FUCKER COMPONENT?

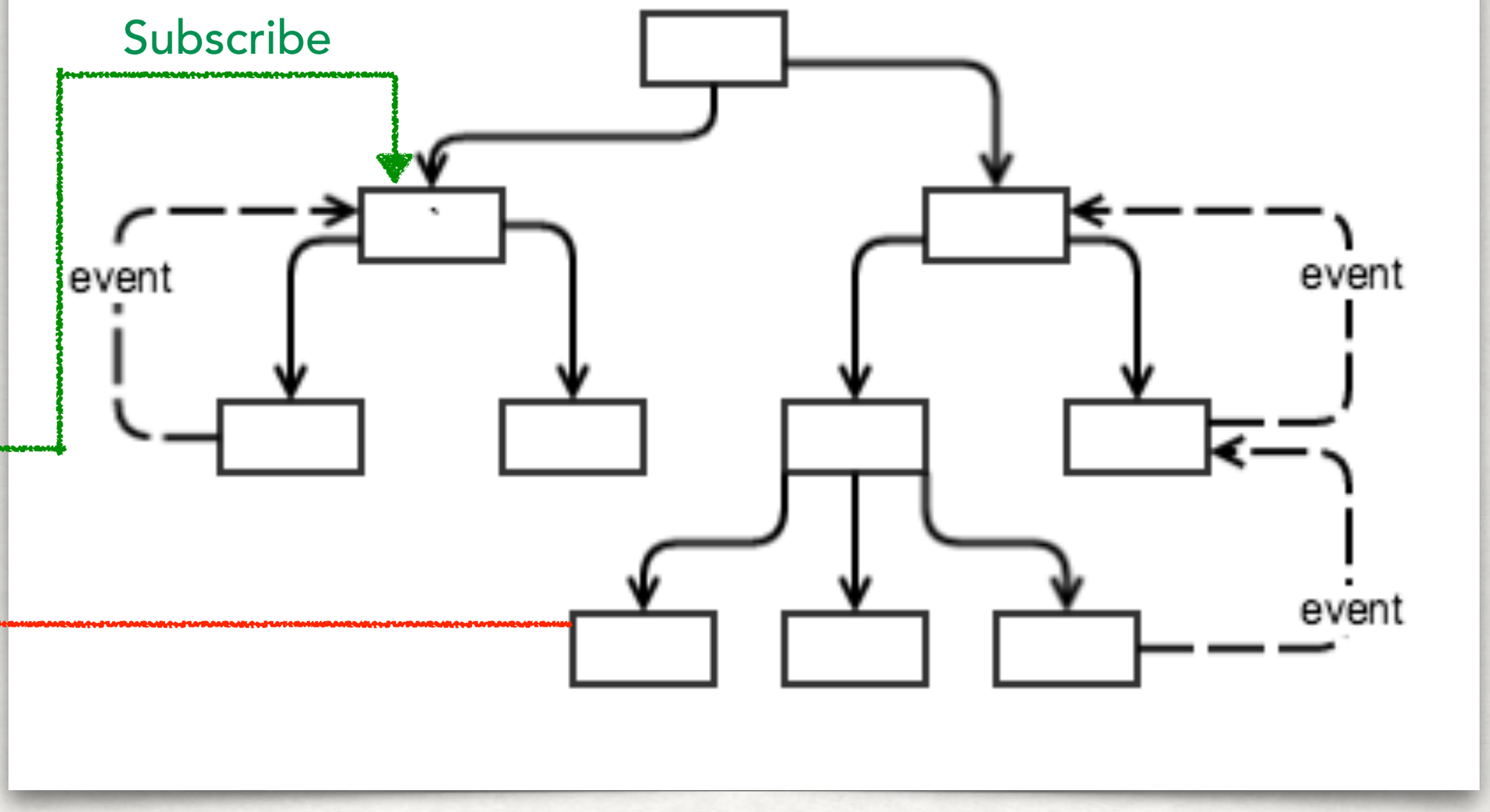

event
event
event

CUSTOM PROVIDERS

CUSTOM PROVIDER

property

Event Emitter



```
export bootstrap(appComponentType: Type,
    customProviders?: Array<any /*Type | Provider | any[]*/>)
```

CREATE SERVICE PROVIDER

SERVICE PROVIDER

- Service Property (What we pass Object.)

```
export class DialogProperty {
    title:string;
    content:any;
    show:boolean = true;
}
```

- Service Event (Pipe for call Component, what we want.)

```
private dialogChange:EventEmitter<DialogProperty> = new EventEmitter();
```

EventEmitter

- emit(value: T)
- next(value: any)
- subscribe(generatorOrNext?: any, error?: any, complete?: any) : any

SERVICE PROVIDER

```
export class DialogProperty {
    title:string;
    content:any;
    show:boolean = true;
}

export class DialogService {
    private dialogChange:EventEmitter<DialogProperty> = new EventEmitter();
    private _dialogProperty:DialogProperty = new DialogProperty();

    open(dialogProperty:DialogProperty) {
        this._dialogProperty = dialogProperty;
        this._dialogProperty.show = true;
        this.emitDialogServiceEvent(this._dialogProperty);
    }

    close() {
        this._dialogProperty.show = false;
        this.emitDialogServiceEvent(this._dialogProperty);
    }

    private emitDialogServiceEvent(dialogProperty:DialogProperty) {
        this.dialogChange.emit(dialogProperty);
    }

    getDialogServiceEmitter() {
        return this.dialogChange;
    }
}
```

EVENT SUBSCRIBER

```
export class Dialog {
  …
    constructor(private dialogService:DialogService) {
        this.dialogSubscription();
    }

  …

    private dialogSubscription() {
        this.subscription =
    this.dialogService.getDialogServiceEmitter()
            .subscribe(item => {
                this.title = item.title;
                this.content = item.content;
                this.showDialog(item.show);
            });
    }

  …
}
```

CUSTOM PROVIDER

property

Event Emitter