ANGULARJS 2 BASIC

Author: Tinh Ha
At: IDEALOGIC

WHAT IS ANGULARJS 2

- Angular 2 is the next version of Google’s massively popular MV* framework for building complex applications in the browser (and beyond).
- Base on Angular 1, but better, more simple & modern
- Angular 2 is used to build web app & mobile app (Nativescript)
- Angular is component/module driven.

ANGULAR 2 CONCEPTS

- Angular is Functional / Component / Module driven.
- Routing & AMD (Asynchronous Module Definition)
- Directive
- Service & Provider
- Dependency Injection
- Property binding
- HTTP & RXJS

COMPONENT BASE DEVELOPMENT

- separation of concerns
- independence, pluggable, reusable
- easier for testing, maintenance

MODULE

```
import { NgModule }      from '@angular/core';
import { BrowserModule } from ‘@angular/platform-browser';
import { SomeModule } from ‘./app-module/SomeModule’;

@NgModule({
    imports: [ BrowserModule, SomeModule ],
    providers: [ Logger ],
    declarations: [ AppComponent ],
    exports: [ AppComponent ],
    bootstrap: [ AppComponent ]
})
export class AppModule { }
```

ROUTE & MODULE LAZY LOADING

```
import { NgModule }             from '@angular/core';
import { Routes, RouterModule } from '@angular/router';

export const routes: Routes = [
    { path: '', redirectTo: 'contact', pathMatch: 'full'},
    { path: 'crisis', loadChildren: 'app/crisis/crisis.module#CrisisModule' },
    { path: 'heroes', loadChildren: 'app/hero/hero.module#HeroModule' }
];

@NgModule({
    imports: [RouterModule.forRoot(routes)],
    exports: [RouterModule]
})
export class AppRoutingModule {}
```

COMPONENT

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

@Component({
    moduleId: module.id,
    selector: 'app-title',
    templateUrl: 'title.component.html',
})
export class TitleComponent {
    @Input() subtitle = '';
    title = 'Angular Modules';
}
```

DIRECTIVE

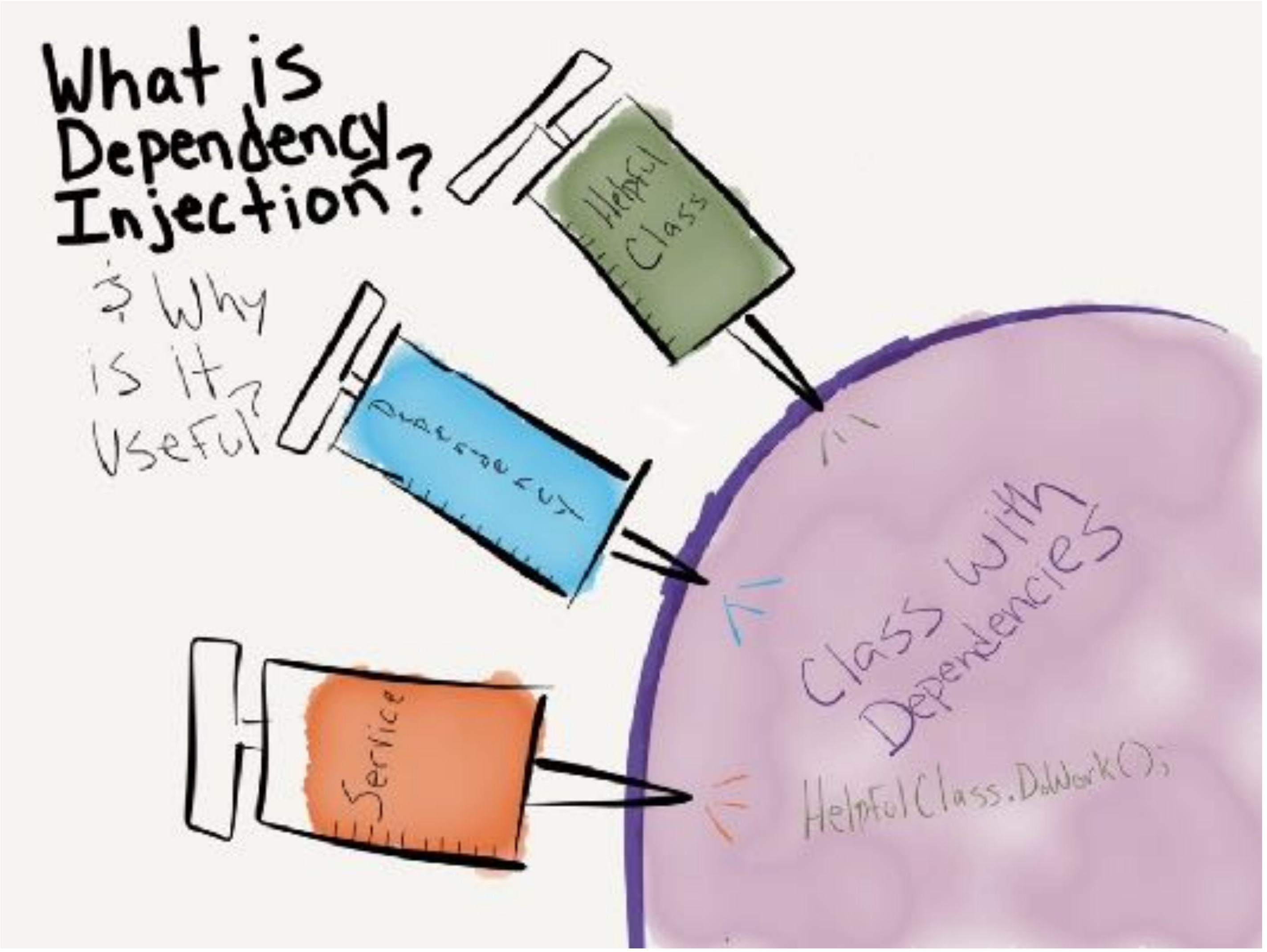
```
import { Directive, ElementRef, Renderer } from '@angular/core';

@Directive({ selector: '[highlight]' })
/** Highlight the attached element in gold */
export class HighlightDirective {
    constructor(renderer: Renderer, el: ElementRef) {
        renderer.setElementStyle(el.nativeElement, 'backgroundColor', 'gold');
    }
}
```

Using directive:

```
  html: <h1 highlight>some title…</h1>
```

DEPENDENCY INJECTION

- dependency injection is a software design pattern that implements inversion of control for resolving dependencies …



SERVICE & PROVIDER

```
import {Injectable} from "@angular/core";

@Injectable()
export class ContactService {
    . . .
 }
```

```
import {ContactService} from “./contact.service";

@NgModule({
    . . .
    providers: [ContactService]
    . . .
})
export class ContactModule { }
```

ZONEJS

AngularJS 2 use ZoneJS for property binding.

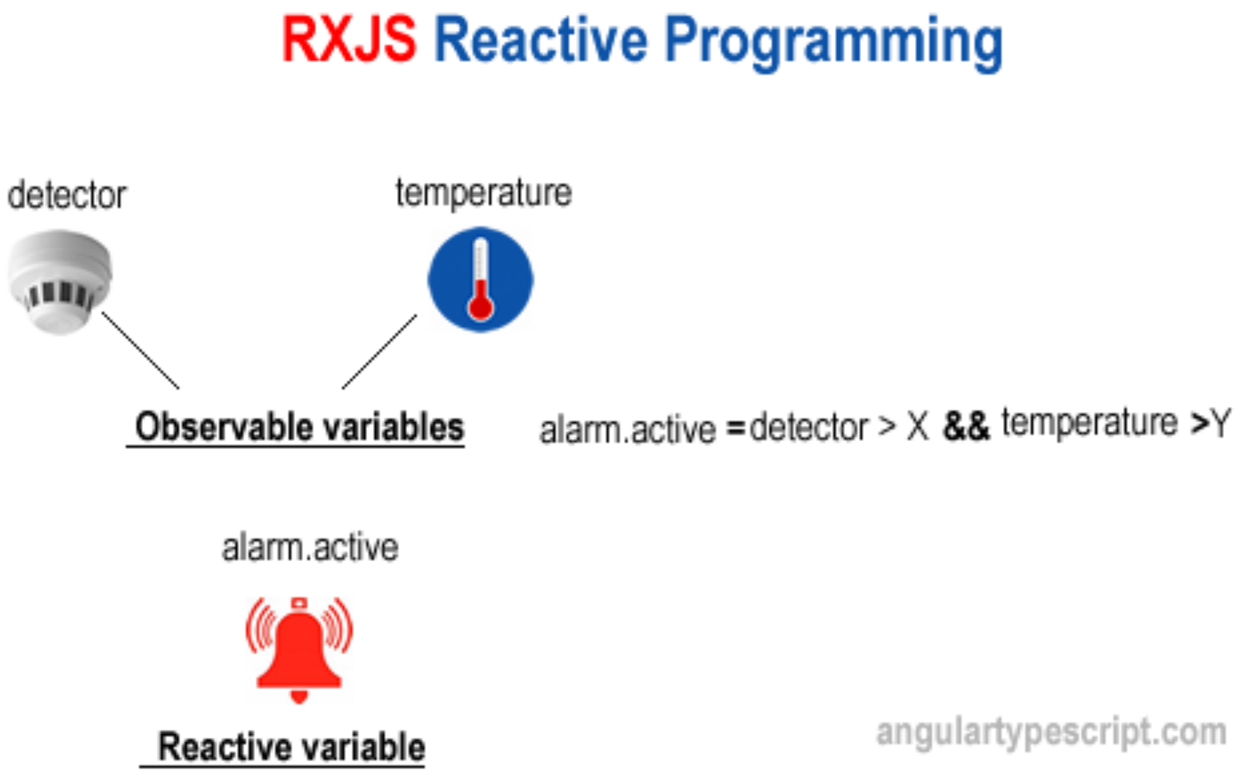
What is ZoneJS: Implements Zones for JavaScript

What is Zone: A Zone is an execution context that persists across async tasks. Basically, it will monkey-patching all the async tasks of browser.

What can ZoneJS do: it monkey-patching every async browser API with zone.wrap.

RXJS

- An API for asynchronous programming
- with observable streams
- Base on Observer & Iterator pattern



HTTP

```
this.heroes = this.searchTerms
    .debounceTime(300)        // wait for 300ms pause in events
    .distinctUntilChanged()   // ignore if next search term is same as previous
    .switchMap(term => term   // switch to new observable each time
          ? this.heroSearchService.search(term)
          : Observable.of<Hero[]>([]))
    .catch(error => {
       …
    });



// Active next flow
this.searchTerms.next(term);
```

THE END

THANKS YOU