ANGULAR 2
CORE CONCEPTS

FABIO **BIONDI** / MATTEO **RONCHI**

unshift.xyz

FABIO**BIONDI**

UI Developer and Trainer

Sviluppo, formazione e consulenza su AngularJS, React, CreateJS, D3.js e diverse librerie Javascript.

fabiobiondi.com

MATTEO**RONCHI**

Senior Software Engineer

Appassionato di architetture e ottimizzazioni da poco aggiunto al team di Workwave

WORKWAVE

@cef62

ANGULARCOMMUNITIES

AngularJS Developer Italiani

AngularJS Italia

ANGULAR 2 **VS 1.X**

- Goodbye $scope
- No more controllers
- Component Based-UI
- 1-way data flow
- ES6 / Typescript
- New built-in directives

ANGULAR 2 **VS 1.X**

- New DI system
- Performance
- Better Mobile Support
- Server side render e Native Script
- Embrace Flux and RxJS
- Change Detection System

COMPONENTFUNDAMENTALS

Component Decorator

Imports

selector name **<tab-bar/>**

```
import {Component} from 'angular2/core';

@Component({
  selector: 'tab-bar',
  template: `<div>...</div>`,
})
export class TabBar {
  // ...
}
```

template

Component Name

CREATE A **WIDGET**

▼ FILES

index.html

src/comps/StaticMap.ts

src/comps/TabBar.ts

src/model/Country.ts

src/Widget.ts

<tab-bar />

<map />

<widget />

Open Plnkr

```
export  class Country {
  constructor(
    public id: string,
    public name: string,
    public coords: string) {
    }
}
```

Automatically generates class properties

`Country.ts` (model)

Selector **<widget/>**

```
@Component({
  selector: 'widget',
  template: `<tab-bar [data]="list"
                      (onTabSelect)="select($event)"></tab-bar>
             <map [item]="country"></map>`,
  directives: [TabBar, StaticMap]
})
export class Widget {
  ...
}
```

Component Injection

Component Name

<widget/> (partial)

```
@Component({
  selector: 'widget',
  template: `<tab-bar [data]="list"
                      (onTabSelect)="select($event)"></tab-bar>
             <map [item]="country"></map>`,
  directives: [TabBar, StaticMap]
})
export class Widget {
  ...
}
```

INPUT PROP

OUTPUT EVENT

INPUT PROP

<widget/> (partial)

```
const countries = [
  new Country( '1', 'Italy' , '42,13' ),
  new Country( '2', 'Greece', '42,25' ),
  new Country( '3', 'Usa', '40.7,-73' ),
];

@Component({
  selector: 'widget',
  template: `<tab-bar [data]="list"
                      (onTabSelect)="select($event)"></tab-bar>
             <map [item]="country"></map>`,
  directives: [TabBar, StaticMap]
})
export class Widget {
  list: Country[] = countries;
  country: Country = new Country();

  select(c:Country) {
    this.country = c;
  }
}
```

1

2

3

<widget/> (completed)

MAP COMPONENT


```
<map [item]="country">
```

[...]
INPUT PROPERTY

```
import {Component, Input} from 'angular2/core';
import { Country } from '../model/Country';

@Component({
  selector: 'map',
  template: `
  <img src="https://maps.googleapis.com/maps/api/staticmap
    ?center={{item.coords}}&zoom=5&size=400x400&..." />`,
})
export class StaticMap {
  @Input()
  item: Country;
}
```

Template Binding

Input property
`item:Country`

<map/>

TABBARCOMPONENT

[...]
INPUT PROPERTY

```
<tab-bar [data]="list"
  (onTabSelect)="doIt($event)">
```

(...)
OUTPUT EVENT

FRAMEWORK DIRECTIVES **ngFor,ngClass**

```
@Component({
  selector: 'tab-bar',
  template: `
  <ul class="nav nav-tabs">
    <li *ngFor="#tab of data"
        [ngClass]="{'active': tab.id === active.id}"
        (click)="onClick(tab)">
      <a>{{tab.label}}</a>
    </li>
  </ul>`,
  inputs: ['data'],
  outputs: ['onTabSelect']
})
```

<tab-bar/>

```
export class TabBar {
    active: Country = {};
    onTabSelect: EventEmitter<Country>;
    constructor() {
        this.onTabSelect = new EventEmitter();
    }
    onClick(tab:Country) {
        this.active = tab;
        this.onTabSelect.emit(tab);
    }
}
```

CURRENT TAB

ASSIGN EMITTER

EMIT EVENT

<tab-bar/>

ANGULARBOOSTRAP

```
ng.bootstrap(src.Widget)
```

1. LOAD LIBRARIES

```
<script src="https://rawgithub.com/systemjs/systemjs/0.19.6/dist/system.js"></script>
<script src="https://code.angularjs.org/tools/typescript.js"></script>
<script src="https://code.angularjs.org/2.0.0-beta.0/angular2-polyfills.js"></script>
<script src="https://code.angularjs.org/2.0.0-beta.0/Rx.js"></script>
<script src="https://code.angularjs.org/2.0.0-beta.0/angular2.dev.js"></script>
```

2. Configure *System.js*

```
<!-- 2. Configure SystemJS -->
<script>
  System.config({
    transpiler: 'typescript',
    typescriptOptions: { emitDecoratorMetadata: true },
    packages: {'src': {defaultExtension: 'ts'}}
  });
</script>

<!-- 3. Bootstrap -->
<script>
  System.import('angular2/platform/browser').then(function(ng){
    System.import('src/Widget').then(function(src) {
      ng.bootstrap(src.Widget);
    });
  });
</script>
</head>

<!-- 4. Display the widget -->
<body>
  <widget class="container" style="display: block">Loading...</widget>
</body>
```

3. Bootstrap

4. DISPLAY **<widget/>**

DEPENDENCY **INJECTION**

NEW **DEPENDENCY INJECTION** ENGINE

- **@injectable** to enable injection to services
- Support multiple providers
- Application level injections
- **Component level** injections

```
import { SubComp } from `./sub-comp`
import { MyHelper } from `./my-helper`

@Component({
   template: `<sub-comp></sub-comp>`
   directives: [SubComp]
})
class MyComp {
   constructor(private helper: MyHelper) {}
}
```

Simple Service

```
export class MyService {

   getData() {
      return loadData.load();
   }
}
```

Inject Service to a Service

```
import {Injectable} from ‘angular2/core’;

@Injectable()
export class MyService {
   constructor(public loadData:LoadData) {}

   getData() {
      return loadData.load();
   }
}
```

COMPONENT **LIFECYCLE**

“ Angular only calls a directive/component hook method if it is defined. “ [docs]

BASE HOOKS

(components & directives)

ngOnChanges	**input** property value changes
ngOnInit	Initialization step
ngDoCheck	every change detection cycle
ngOnDestroy	before destruction

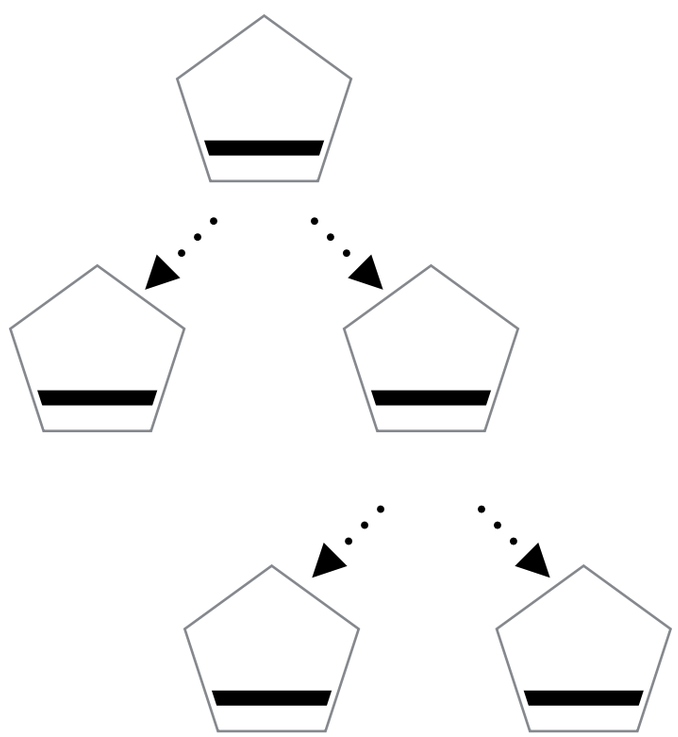
```
@Directive({selector: '[my-spy]'})
class Spy implements OnInit, OnDestroy {
  ngOnInit() {
     console.log(`onInit`);
   }
  ngOnDestroy() {
     console.log(`onDestroy`);
   }
}
```

Usage: `<div my-spy>...</div>`

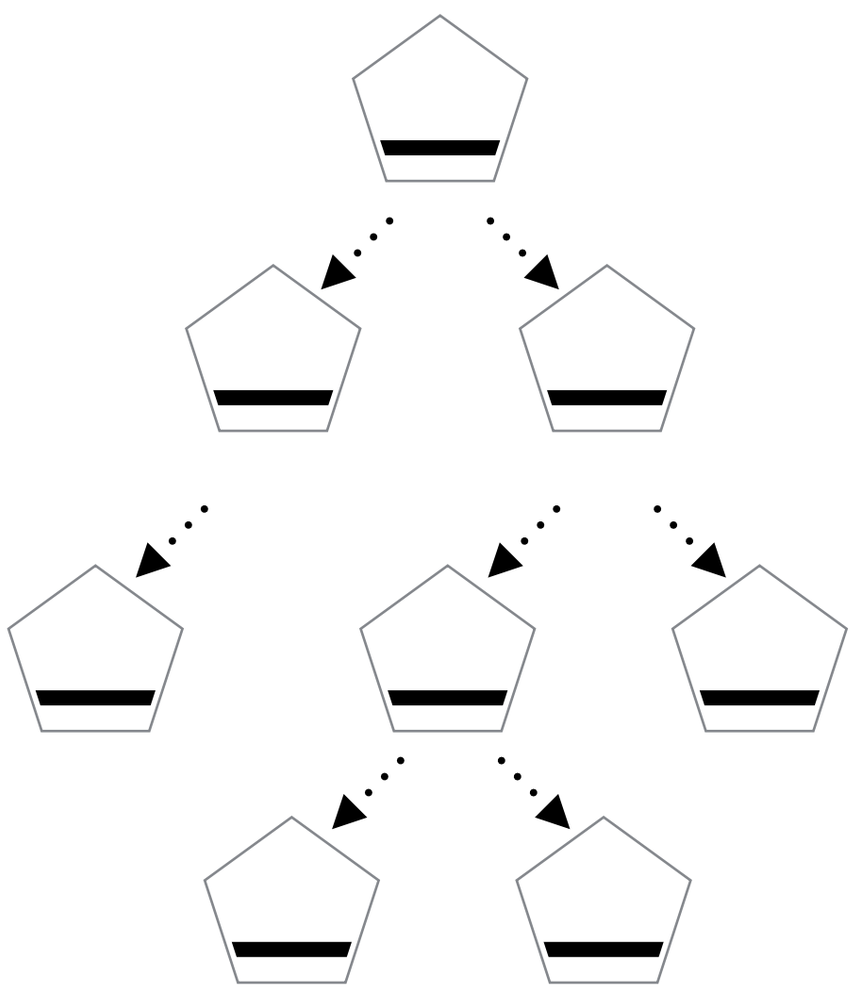
CHANGE DETECTION

Angular Application are Data Driven

Data Model

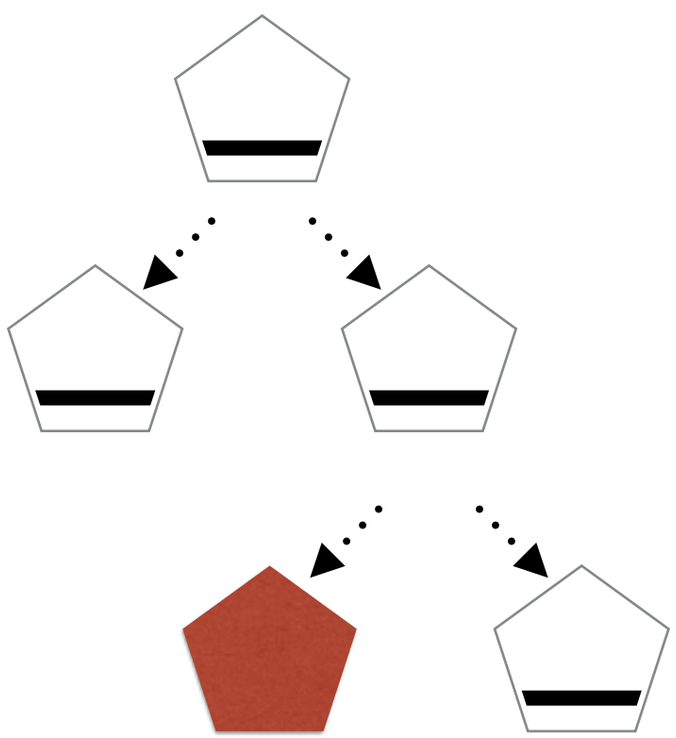


Components

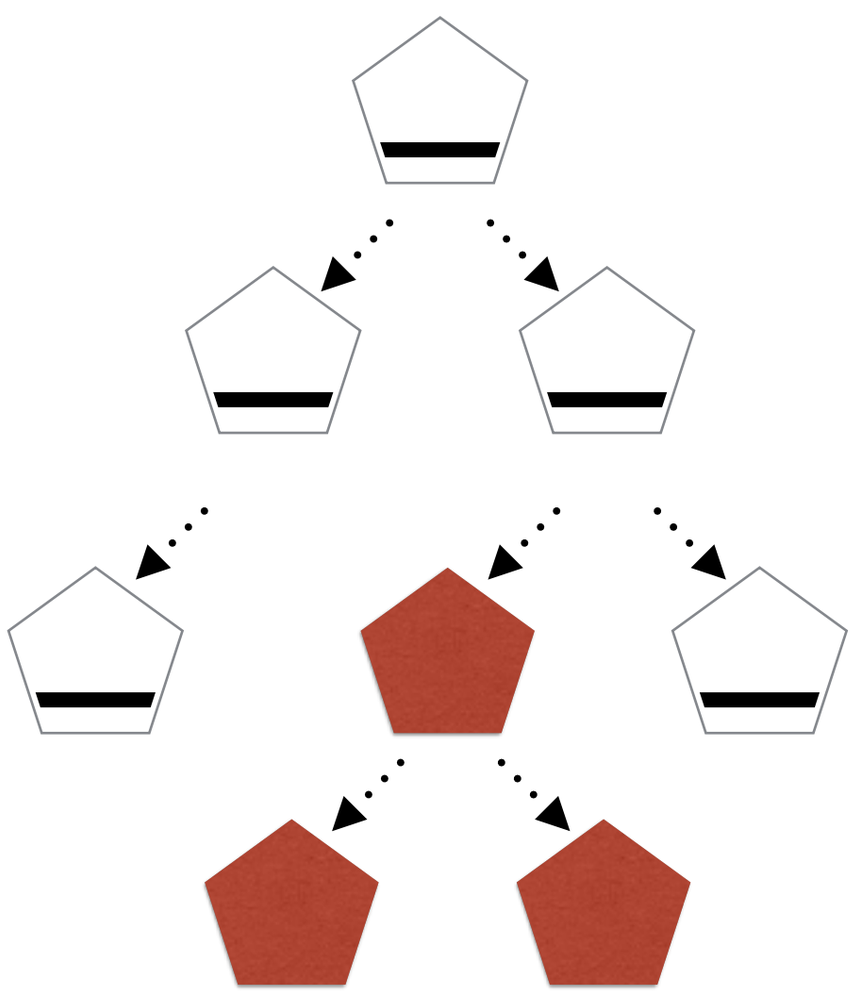


DATA CHANGES -> VIEW UPDATES

Data Model

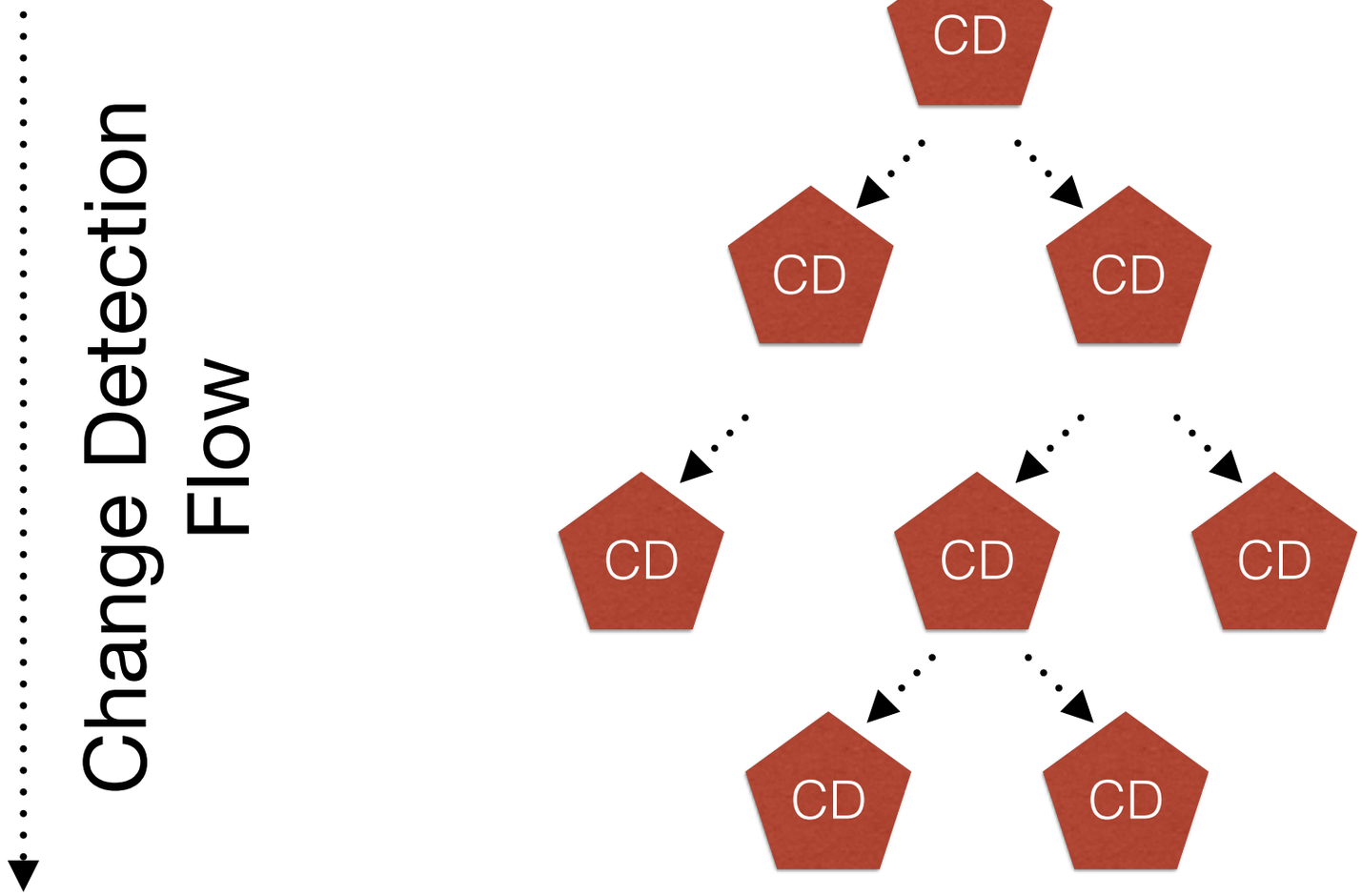


Components



CHANGE DETECTION

TRAVELS TOP TO BOTTOM



CHANGE DETECTION IS DEFINED AT **COMPONENT** LEVEL

CHANGE DETECTION

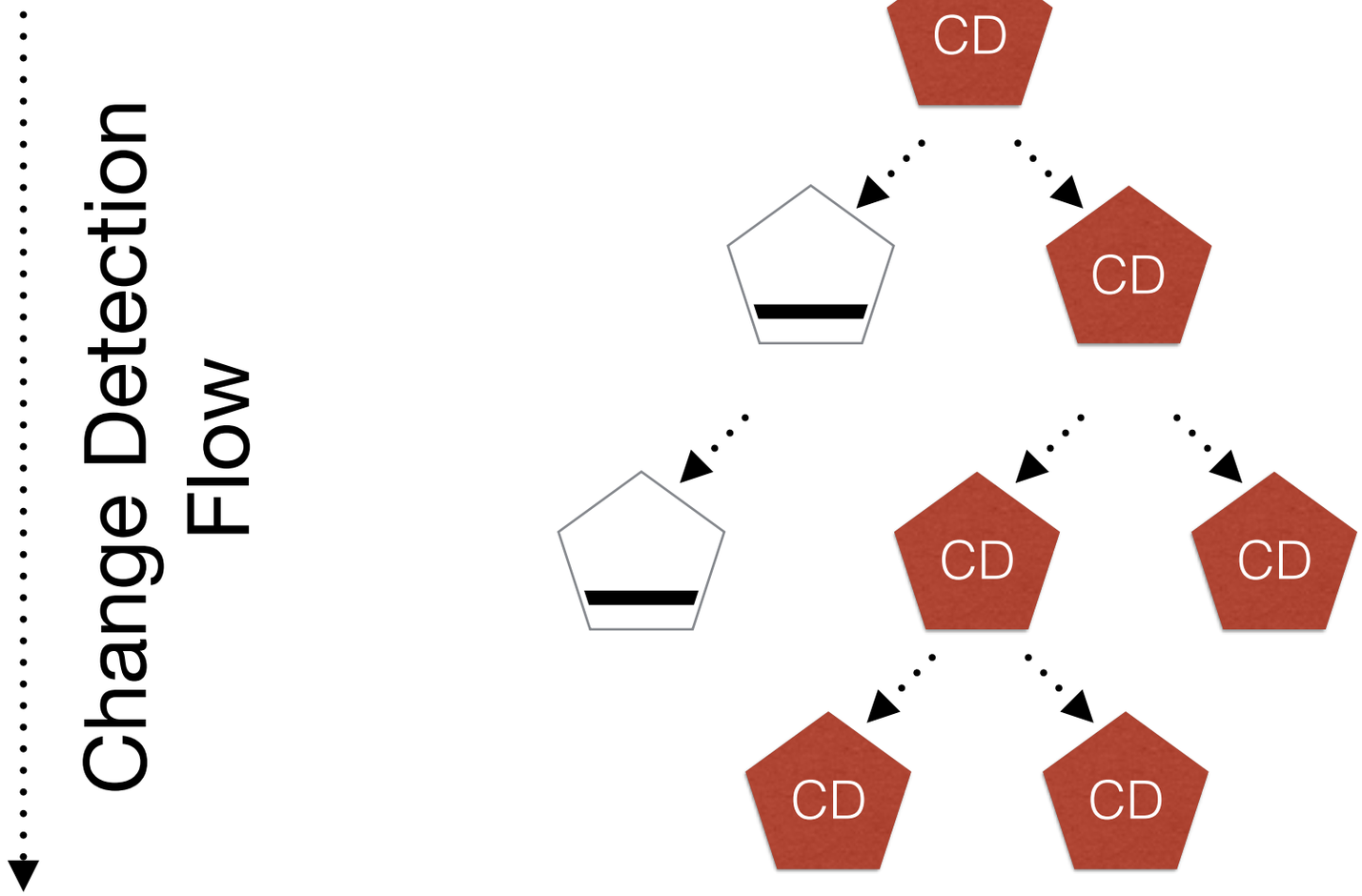
~~CAN~~ **SHOULD** BE OPTIMIZED

- Immutable Data
- Observable
- Custom BUS Systems …

Enable Smart Change Detection

```
@Component({
    template: `
        <h1>{{user.name}}</h1>
        <h3>{{user.nickName}}</h3> `,
        changeDetection: ChangeDetectionStrategy.OnPush
        inputs: [user]
})
class MyComp {}
```

CHANGE DETECTION
WITH **IMMUTABLE DATA**

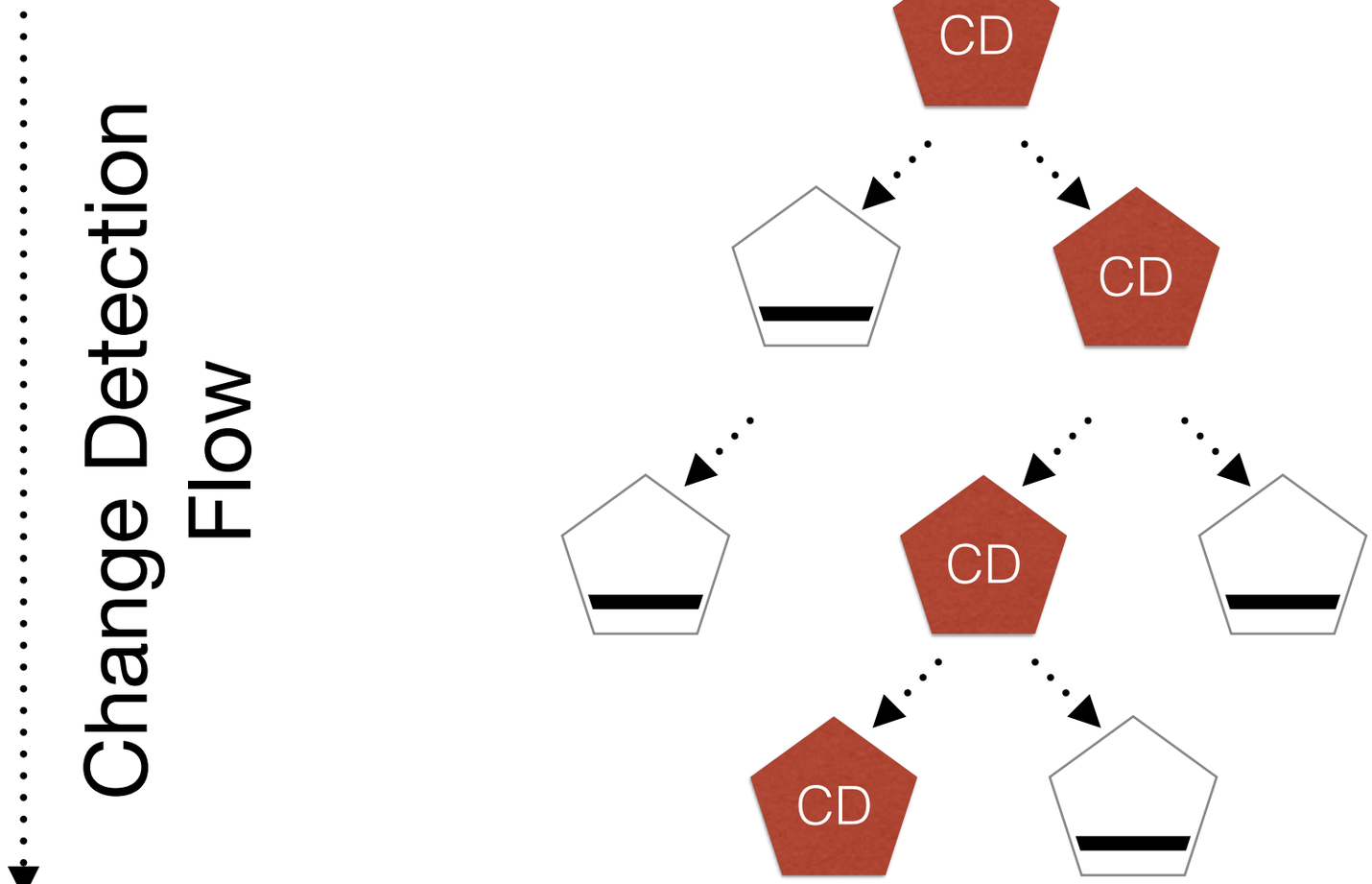


Change Detection with Observable

```
@Component({
    template: `
        <h1>{{user.name}}</h1>
        <h3>{{user.nickName}}</h3> `,
    changeDetection: ChangeDetectionStrategy.OnPush
})
class MyComp {
    @Input() user$:Observable<User>;
    constructor(private detector: ChangeDetectorRef) {}

    ngOnInit() {
        this.user$.subscribe((user) => {
            this.user = user;
            this.detector.markForCheck();
        })
    }
}
```

CHANGE DETECTION
WITH OBSERVABLES



WHAT **CAUSE**
CHANGE DETECTION

- setTimeout(), setInterval()
- User **Events** (click, input change..)
- **XHR** Requests

GET IN THE **ZONE**

ZONE.JS INTERCEPTS ALL **ASYNC** OPERATIONS

Angular has its own **NgZone** to controls **Change Detections**

THANKS!

FABIO **BIONDI** / **fabiobiondi.com**

MATTEO **RONCHI** / **@cef62**