

Introduction into Angular 2

Victor Savkin



Great for Building
Large Applications

And Easy to Get
Started With

```
1 import {Component, View, CoreDirectives, EventEmitter, bootstrap} from 'angular2/angular2';
2
3
```

```
@Component({
  selector: 'talk-cmp',
  properties: ['talk'],
  events: ['rate']
})
@View({
  directives: [CoreDirectives],
  template: `
    Rating {{talk.avgRating}} {{talk.name}}
    <span *if="talk.speaker">Speaker: {{talk.speaker}}</span>
    <button (click)="triggerRate()">Rate</button>
  `
})
class TalkCmp {
  constructor() {
    this.rate = new EventEmitter();
  }

  triggerRate() {
    var rating = 1 + Math.floor(Math.random() * 9);
    this.rate.next({rating: rating});
  }
}
```

```
@Component({
  selector: 'talks-app'
})
@View({
  directives: [CoreDirectives, TalkCmp],
  template: `
    <h2>Talks</h2>
    <talk-cmp *for="var t of talks" [talk]="t" (rate)="rateTalk(t, $event.rating)"></talk-cmp>
  `
})
class TalksApp {
  constructor() {
    this.talks = [
      { name: 'Are we there yet?', speaker: 'Rich Hickey', avgRating: 0, ratings: [] },
      { name: 'The value of values', speaker: 'Rich Hickey', avgRating: 0, ratings: [] },
      { name: 'Simple Made Easy', speaker: null, avgRating: 0, ratings: [] }
    ]
  }

  rateTalk(talk, rating) {
    talk.ratings.push(rating);
    talk.avgRating = Math.round(talk.ratings.reduce((a, b) => a + b) / talk.ratings.length);
  }
}

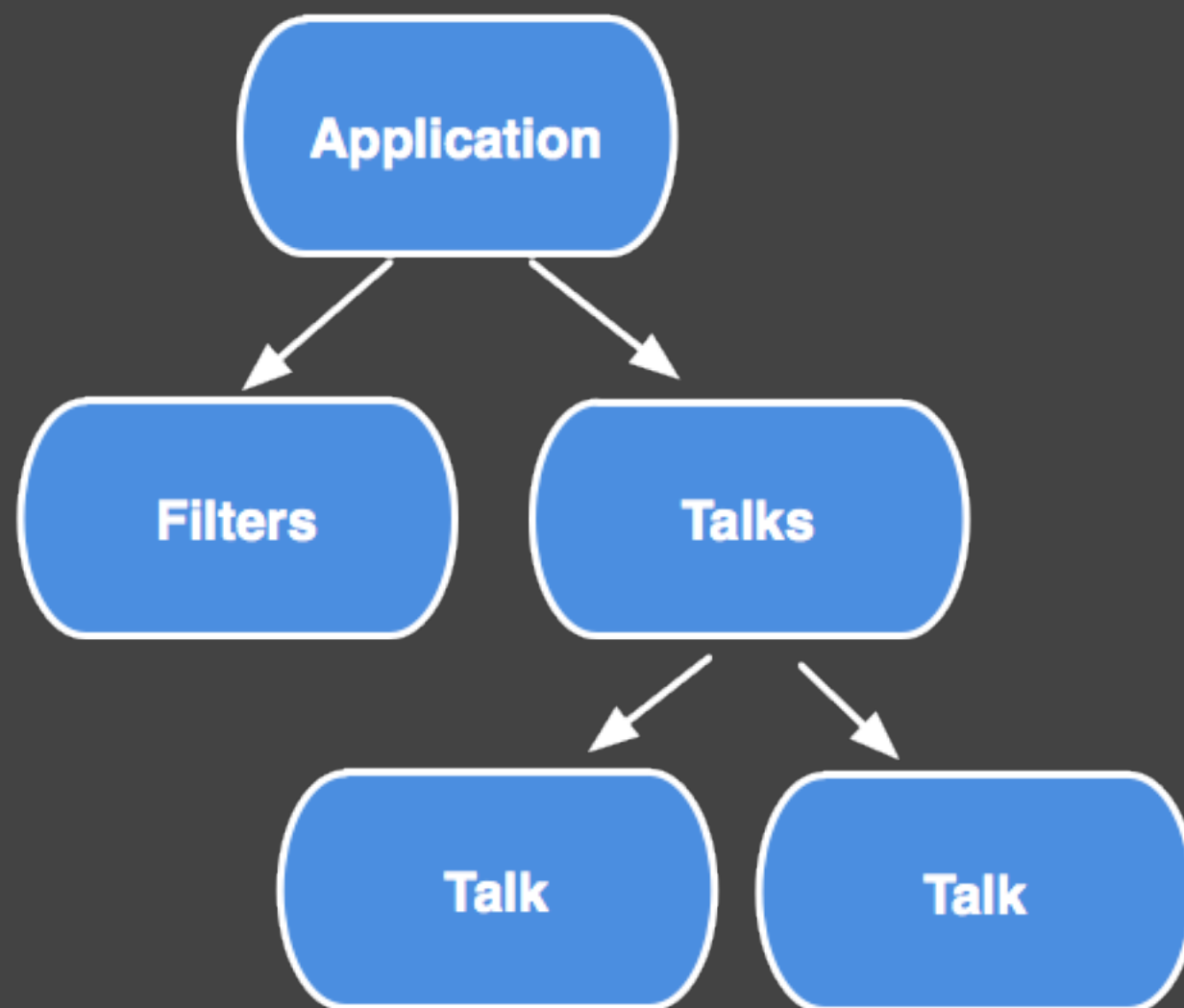
export function main() {
  bootstrap(TalksApp);
}
```

Components

Dependency
Injection

Property
Bindings

Components



Speaker

Rich Hickey

FILTER

Rating
9.1

Are We There Yet?

Rich Hickey

WATCH

RATE

Rating
8.5

The Value of Values

Rich Hickey

WATCH

RATE

Rating
8.2

Simple Made Easy

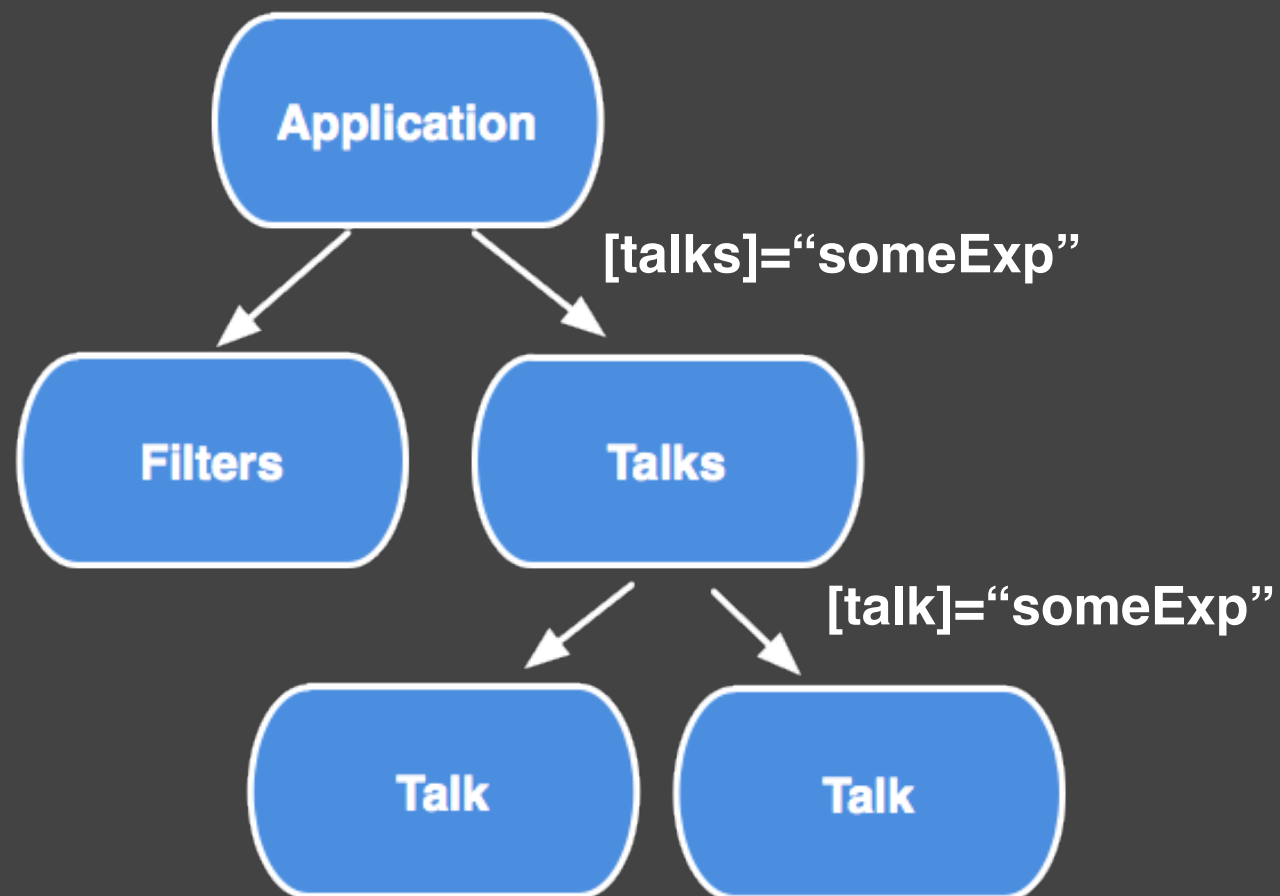
Rich Hickey

WATCH

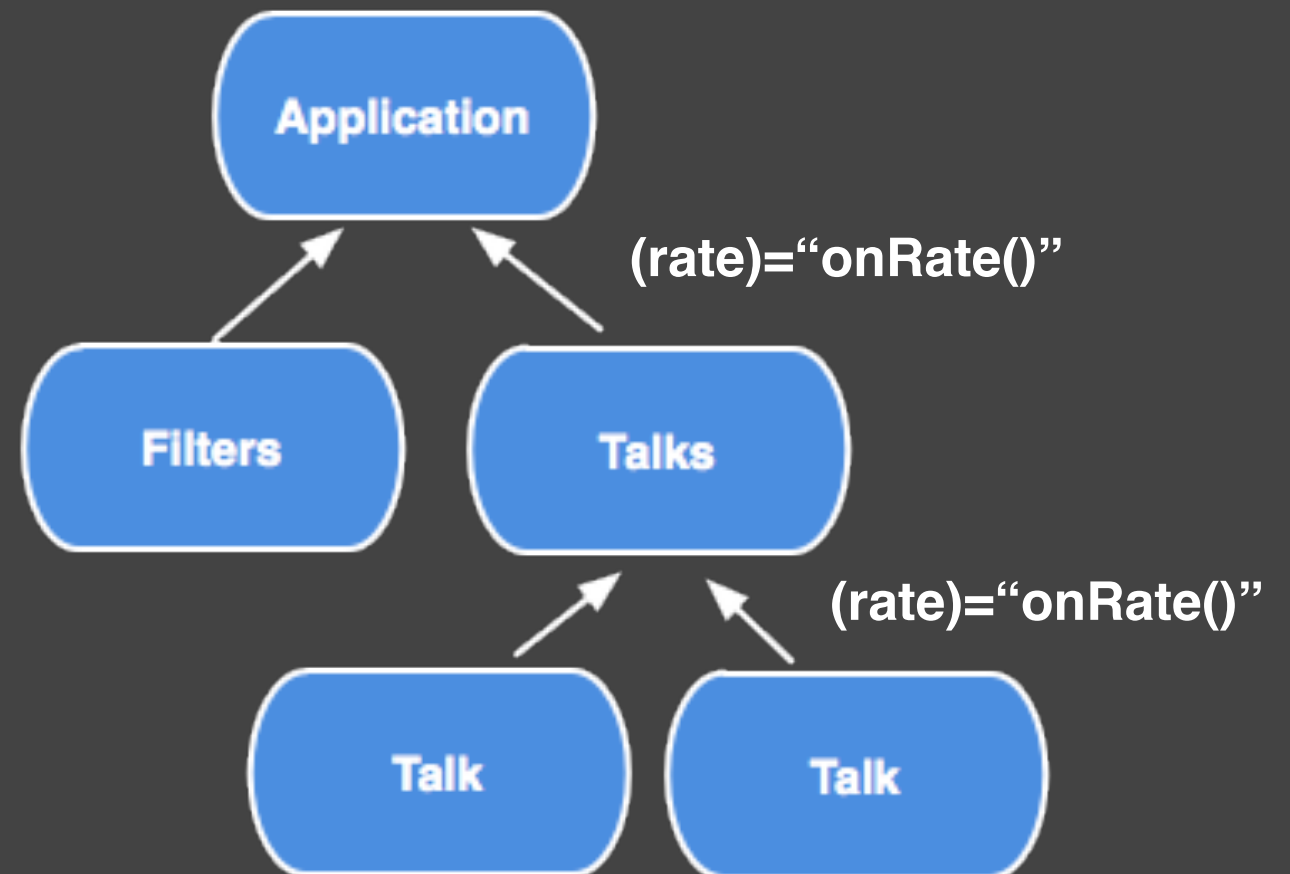
RATE

```
@Component({
  selector: 'talk-cmp',
  properties: ['talk'],
  events: ['rate'],
  lifecycle: []
})
@View({
  template: `
    {{talk.title}}
    {{talk.speaker}}
    <formatted-rating [rating]="talk.rating"></formatted-rating>
    <watch-button [talk]="talk"></watch-button>
    <rate-button [talk]="talk"></rate-button>
  `
})
class TalkCmp {
  talk: Talk;
  rate: EventEmitter;
  //...
}
```

Parent -> Child



Child -> Parent



```
<talk-cmp [talk]="someExp" (rate)="onRate()"></talk-cmp>
```

```
@Component({
  selector: 'talk-cmp',
  properties: ['talk'],
  events: ['rate'],
  lifecycle: []
})
```

```
@View({
  template: `
    {{talk.title}}
    {{talk.speaker}}
    <formatted-rating [rating]="talk.rating"></formatted-rating>
    <watch-button [talk]="talk"></watch-button>
    <rate-button [talk]="talk"></rate-button>
  `
})
```

```
class TalkCmp {
  talk: Talk;
  rate: EventEmitter;
  //...
}
```

```
@Component({
  selector: 'talk-cmp',
  properties: ['talk'],
  events: ['rate'],
  lifecycle: []
})
@View({
  template: `
    {{talk.title}}
    {{talk.speaker}}
    <formatted-rating [rating]="talk.rating"></formatted-rating>
    <watch-button [talk]="talk"></watch-button>
    <rate-button [talk]="talk"></rate-button>
  `
})
class TalkCmp {
  talk: Talk;
  rate: EventEmitter;
  //...
}
```

```
@Component({  
  selector: 'cares-about-changes',  
  lifecycle: [onChange]  
})  
class CareAboutChanges {  
  onChange(changes) {  
    //..  
  }  
}
```

```
@Component({
  selector: 'conf-app',
  injectables: [ConfAppBackend, Logger]
})
class TalksApp {
  //...
}

class TalksCmp {
  constructor(backend: ConfAppBackend) {
    //...
  }
}
```



```
@Component({
  selector: 'input[trimmed]',
  hostListeners: {input: 'onChange($event.target.value)'},
  hostProperties: {value: 'value'}
})
class TrimmedInput {
  value: string;
  onChange(updatedValue: string) {
    this.value = updatedValue.trim();
  }
}
```

Components are
Self-Describing

Used Directly

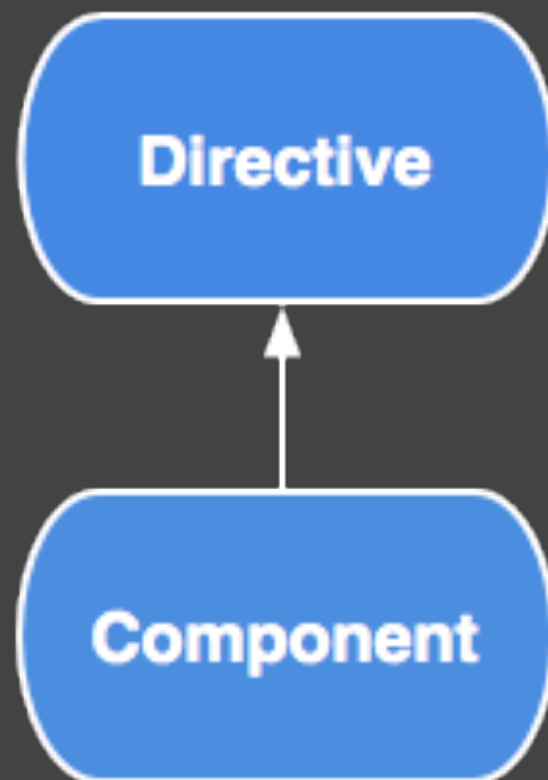
```
<conf-app></conf-app>
```

As an Application or Route

```
bootstrap(ConfApp);
```

What Happened to
Directives?

Components are Directives

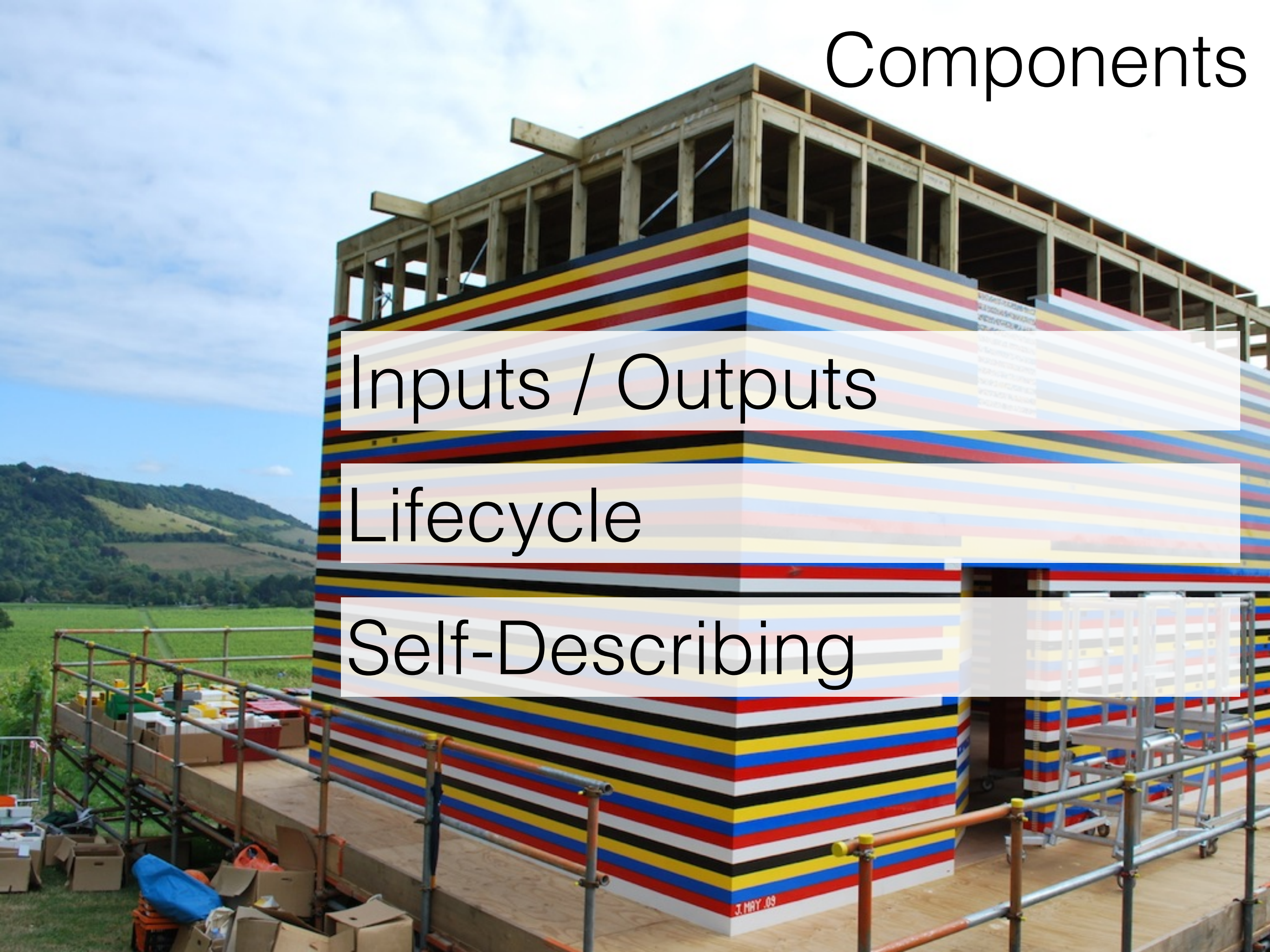


Components

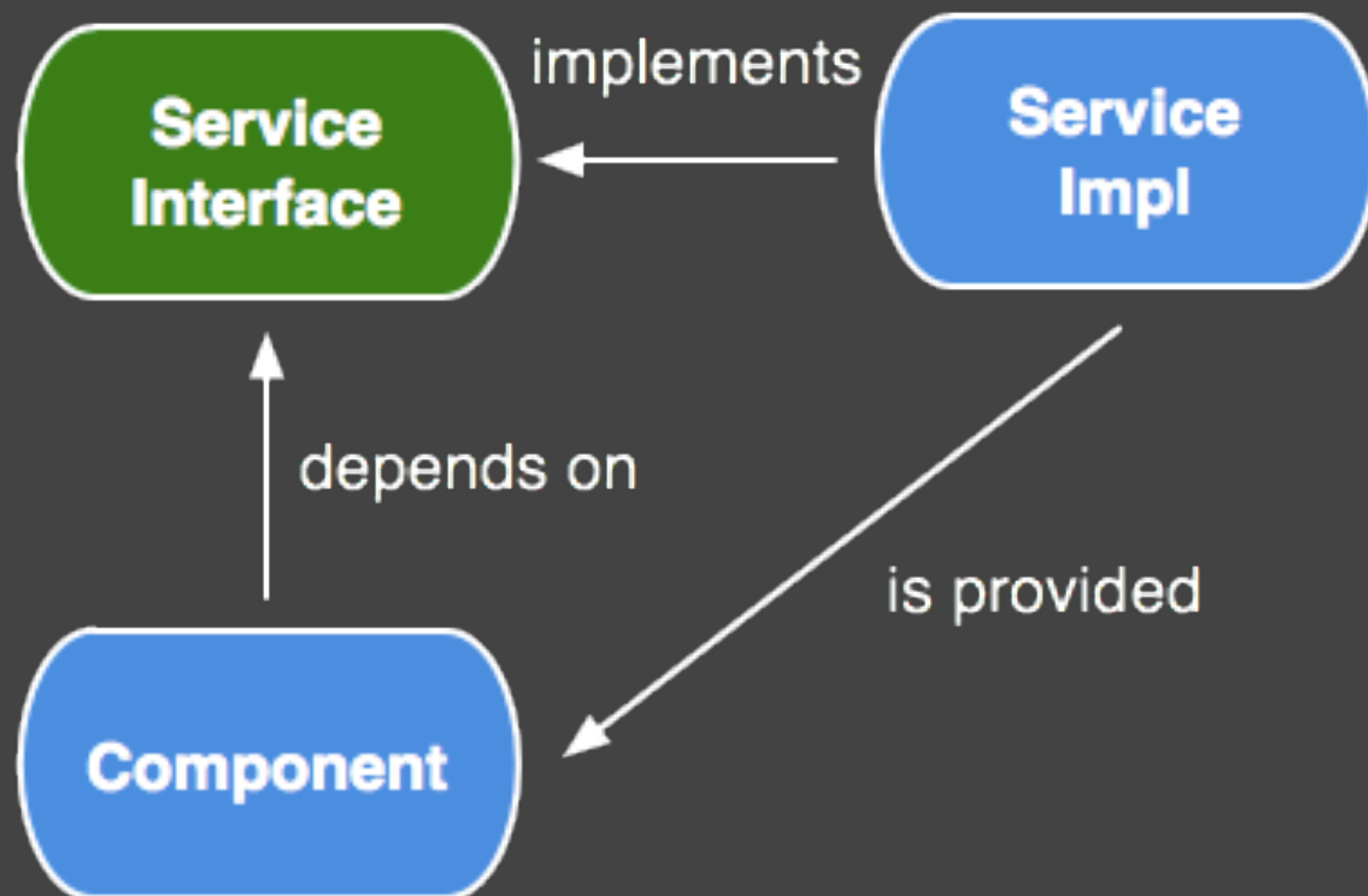
Inputs / Outputs

Lifecycle

Self-Describing



Dependency Injection

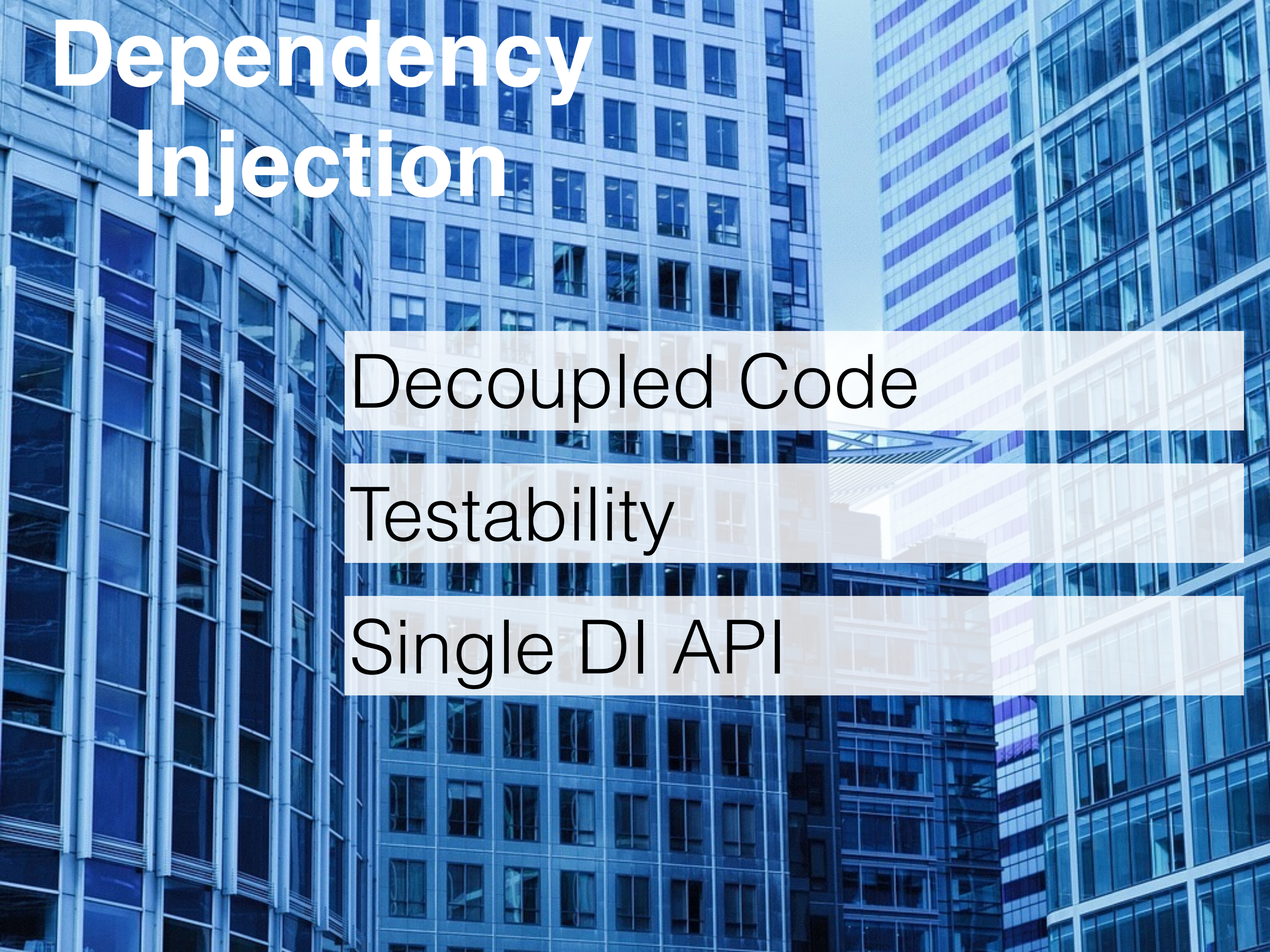



```
16 @Component({
17   selector: 'talks-list'
18 })
19 @View({
20   directives: [CoreDirectives],
21   template: `
22     <h2>Talks:</h2>
23     <div *for="var t of talks">
24       {{t.name}}
25     </div>
26   `
27 })
28 class TalksList {
29   constructor() {
30     // get talks
31   }
32 }
33
34 @Component({
35   selector: 'talks-app'
36 })
37 @View({
38   directives: [TalksList],
39   template: `
40     <talks-list></talks-list>
41   `
42 })
43 class TalksApp {
44 }
45
46 export function main() {
47   bootstrap(TalksApp);
48 }
```

```
16 @Component({
17     selector: 'talks-list'
18 })
19 @View({
20     directives: [CoreDirectives],
21     template: `
22         <h2>Talks:</h2>
23         <div *for="var t of talks|async">
24             {{t.name}}
25         </div>
26     `
27 })
28 class TalksList {
29     constructor(backend: TalksAppBackend) {
30         this.talks = backend.fetchTalks();
31     }
32 }
33
34 @Component({
35     selector: 'talks-app',
36     injectables: [TalksAppBackend]
37 })
38 @View({
39     directives: [TalksList],
40     template: `
41         <talks-list></talks-list>
42     `
43 })
44 class TalksApp {
45 }
```

```
class TalksCmp {  
  constructor(elRef:ElementRef, backend:ConfAppBackend) {  
  }  
}
```

```
class Component {  
    constructor(sibling:SiblingCmp,  
                @Parent parent:ParentCmp,  
                @Ancestor ancestor:AncestorCmp) {  
    }  
}
```

Dependency Injection

Decoupled Code

Testability

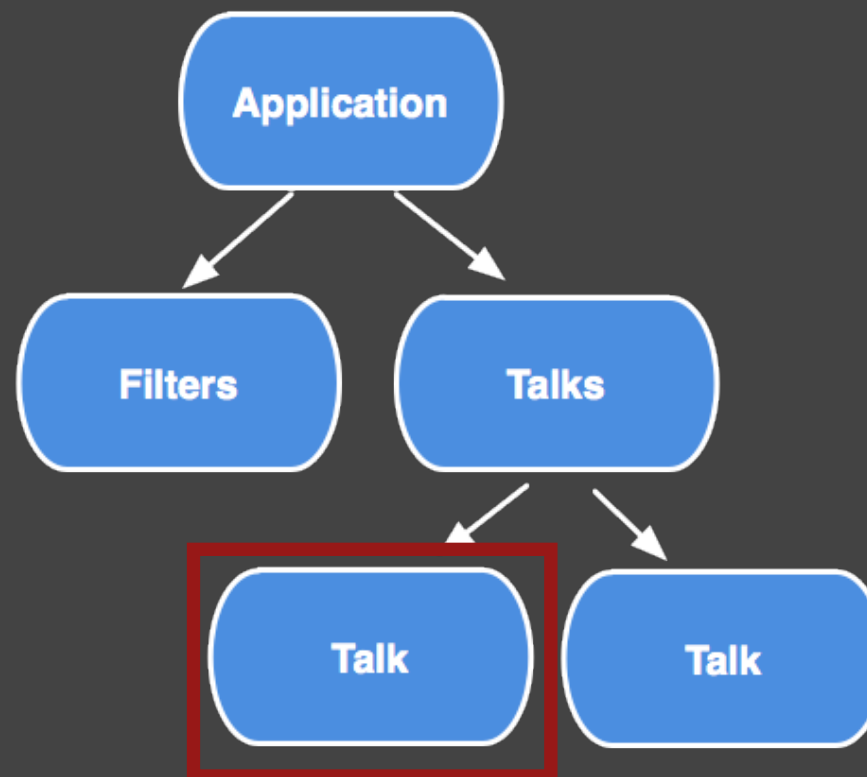
Single DI API

Property Bindings

```

{
  "filters": {
    "speaker": "Rich Hickey"
  },
  "talks": [
    {
      "title": "Are We There Yet?",
      "speaker": "Rich Hickey",
      "yourRating": 9.9,
      "rating": 9.1
    }
  ]
}

```



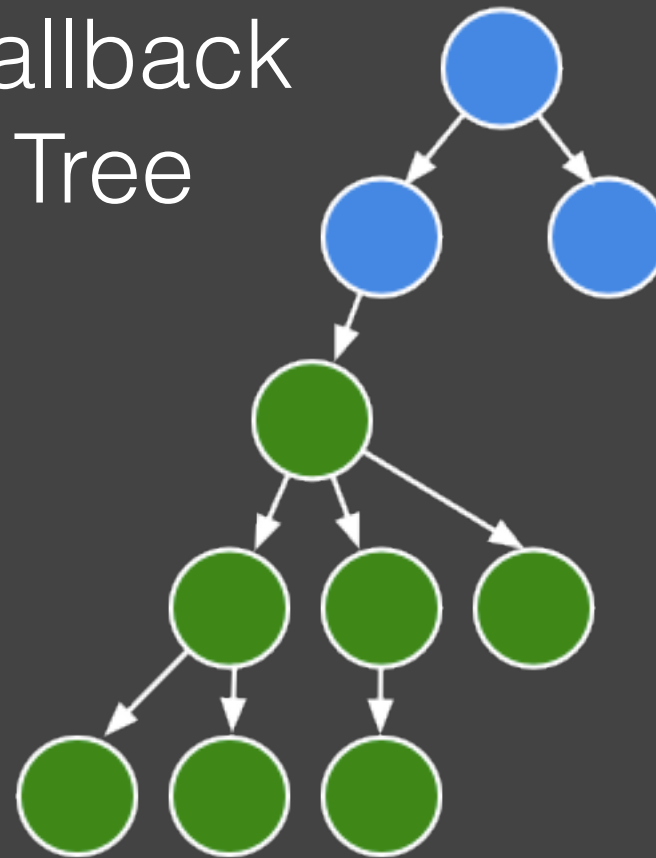
Speaker	
Rich Hickey	
FILTER	
Rating 9.1	Are We There Yet? Rich Hickey WATCH RATE
Rating 8.5	The Value of Values Rich Hickey WATCH RATE
Rating 8.2	Simple Made Easy Rich Hickey WATCH RATE

Model ➡ Components ➡ DOM


```
1 import {Component, View, bootstrap, EventEmitter, CoreDirectives} from 'angular2/angular2'
2
3
```

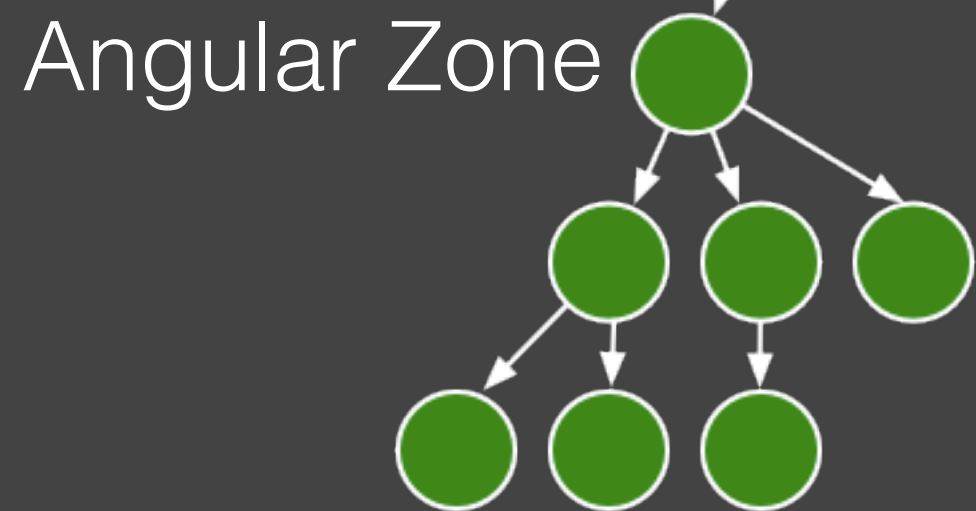
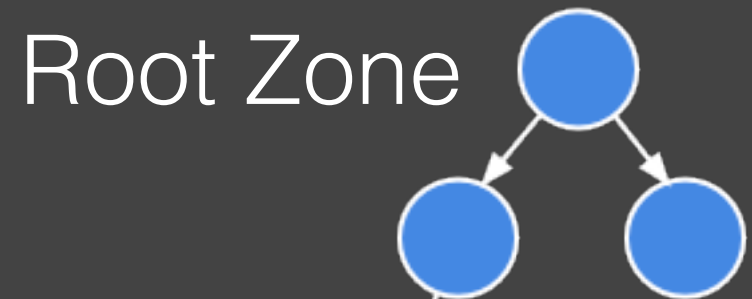
```
3 @Component({
4   selector: 'change-detection-app'
5 })
6 @View({
7   directives: [CoreDirectives],
8   template: `
9     <div *for="var item of items">{{item}}</div>
10 `
11 })
12 class ChangeDetectionApp {
13   constructor() {
14     this.items = ['Item 1', 'Item 2'];
15
16     setInterval(() => {
17       this.items.push(`Item ${this.items.length + 1}`);
18     }, 3000);
19   }
20 }
21
22 export function main() {
23   bootstrap(ChangeDetectionApp);
24 }
```

Callback
Tree



Execution
Order





No More
scope.\$apply

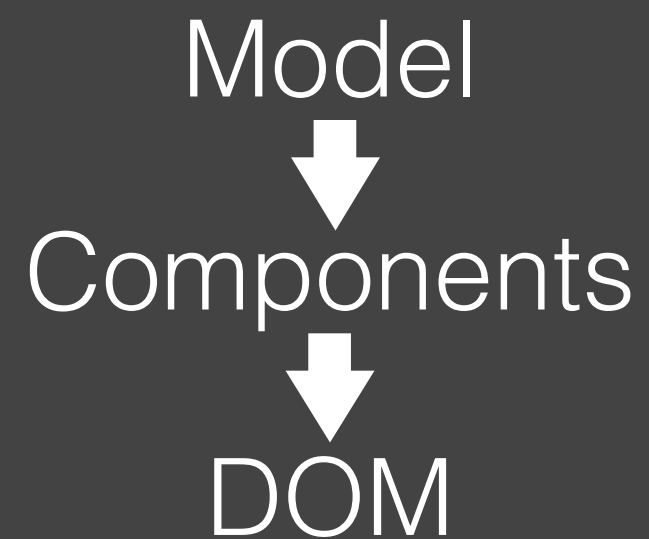
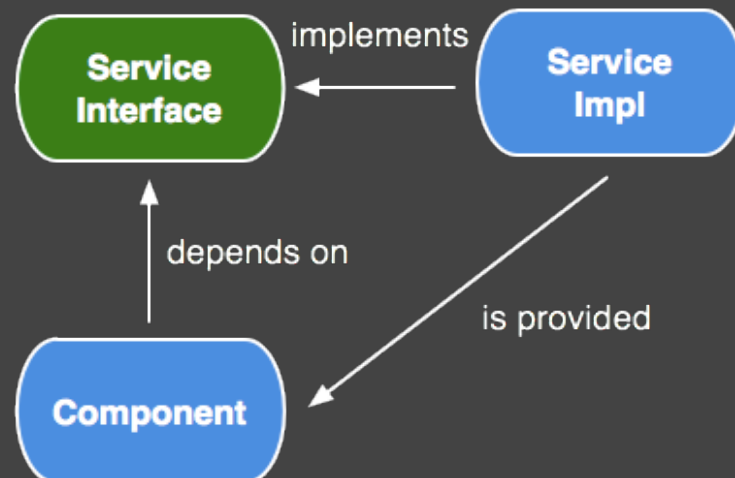
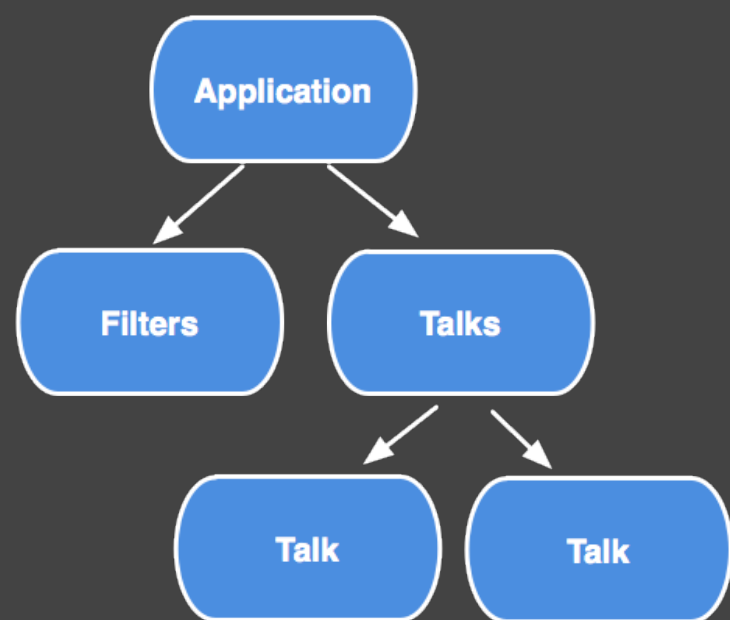
Property Bindings and Zones

Syncs up Model and
Component Tree

Syncs up Component Tree
and the DOM

Uses Zones to know when
to do it

Summary



- Form Handling
- Data Access
- Router
- Animations
- Material Components
- Unit Testing
- E2E Testing



Components

The diagram shows a hierarchical structure of components. At the top is a box labeled 'Application'. Two arrows point from 'Application' to two boxes labeled 'Filter' and 'Talk'. From the 'Talk' box, two arrows point to two more boxes, both labeled 'Talk'.



Dependency Injection

The diagram illustrates the relationship between a service and its consumers. At the top, a box labeled 'Service Interface' has an arrow labeled 'implements' pointing to a box labeled 'Service Impl'. Below 'Service Interface', a box labeled 'Component' has an arrow labeled 'depends on' pointing to 'Service Interface'. Another arrow labeled 'is provided' points from 'Service Impl' to 'Component'.



Property Bindings & Zones

The diagram shows a flow from 'Model' at the top to 'DOM' at the bottom. A box labeled 'Components' is in the middle. Arrows point from 'Model' to 'Components' and from 'Components' to 'DOM'. A large blue arrow points downwards through the center of the diagram.

Learn More

Angular 2 Step by Step Guide

<https://angular.io/docs/js/latest/guide/>

Brian Ford on Zones

https://www.youtube.com/watch?v=3lqtmUscE_U

Victor Savkin's Blog

<http://victorsavkin.com>

Thank You!

@victorsavkin