New Data Architecture in Angular 2

by Gerard Sans | @gerardsans

GDG DevFest Ukraine 2016

A little about me

a bit more...

AngularConnect - 27-28th Sept

@AngularConnect

Angular 2

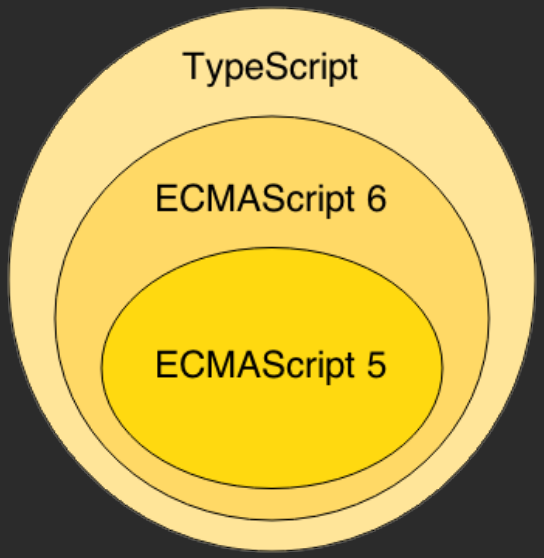
Features

- Latest Web Standards
- Simple
- Lightning fast
- Works everywhere

builtwithangular2.com

ES5, ES6 and TypeScript

ES5 / ES6 / TypeScript



ES6 (ES2015)

- Classes, modules, arrow functions

TypeScript

- Types, annotations, generics, interfaces
- Great editor support

Angular 2 Tooling

Angular 2 Tooling

Bootstrapping

Bootstrapping

- Angular Application instantiation
- Root Module (AppModule)
- Global Dependencies
  - Router, Http, Services
  - Global Values
  - Vendor dependencies

index.html

```
<!DOCTYPE html>
<html>
  <head>
    <!-- Polyfill(s) for older browsers -->
    <script src="https://unpkg.com/core-js/client/shim.min.js"></scri
    <script src="https://unpkg.com/zone.js@0.6.17?main=browser"></scr
    <script src="https://unpkg.com/reflect-metadata@0.1.3"></script>
    <script src="https://unpkg.com/systemjs@0.19.27/dist/system.src.j
    <script src="systemjs.config.js"></script>
    <script>System.import('app');</script>
  </head>
  <body>
    <my-app>
      Loading...
    </my-app>
  </body>
</html>
```

main.ts

```
import {platformBrowserDynamic} from '@angular/platform-browser-
import {AppModule} from './app';

platformBrowserDynamic().bootstrapModule(AppModule)
```

app.module.ts

```
import { NgModule } from '@angular/core';
import { BrowserModule } from '@angular/platform-browser';
import { App } from './app.component';

@NgModule({
  imports: [ BrowserModule ],
  declarations: [ App ],
  bootstrap: [ App ]
})
export class AppModule {}
```

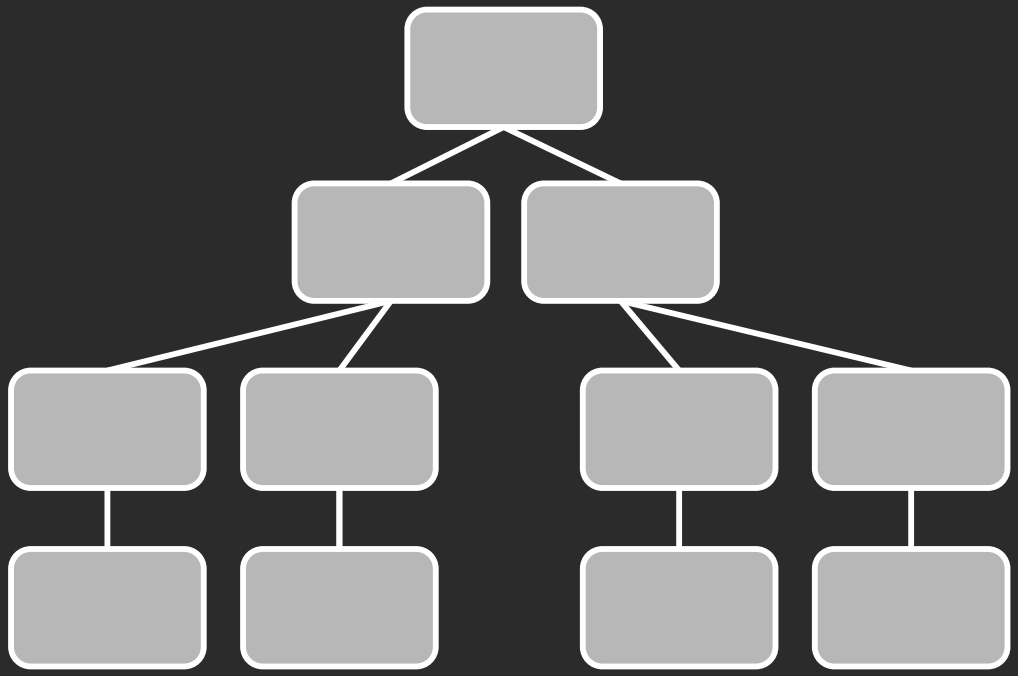
app.component.ts

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

@Component({
  selector: 'my-app', // <my-app>Loading...</my-app>
  template: `...`
})
export class App {
  constructor() { }
}
```

Components

Components Tree



source: blog

Component

- @Component annotation
- Communications
  - Inputs, @Input
  - Outputs, @Output
- Component Lifecycle Hooks
- Host element interaction

Component

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

@Component({
  selector:     'home', // <home></home>
  styles:       [`h1 { color: red }`],
  template:     `<h1>Home</h1>`
})
export class Home { ... }
```

Lifecycle Hooks

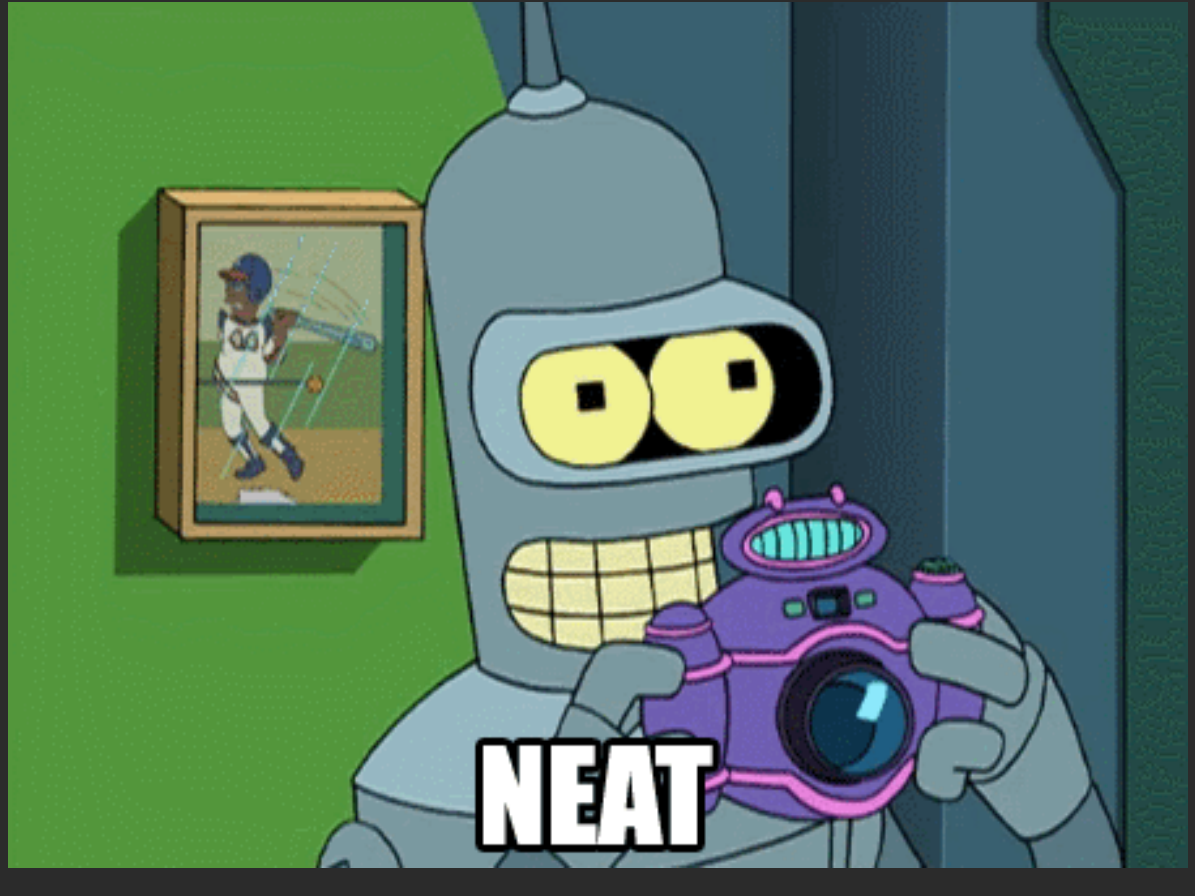
```
import { Component, OnChanges, OnInit, OnDestroy } from '@angula

@Component()
export class myComponent implements OnChanges, OnInit, OnDestroy
  /* 1 */ constructor() { }

  // called when an input or output binding changes
  /* 2 */ ngOnChanges(changes) { }

  // after child initialisation
  /* 3 */ ngOnInit() { }

  // just before is destroyed
  /* 4 */ ngOnDestroy() { }
}
```


NEAT

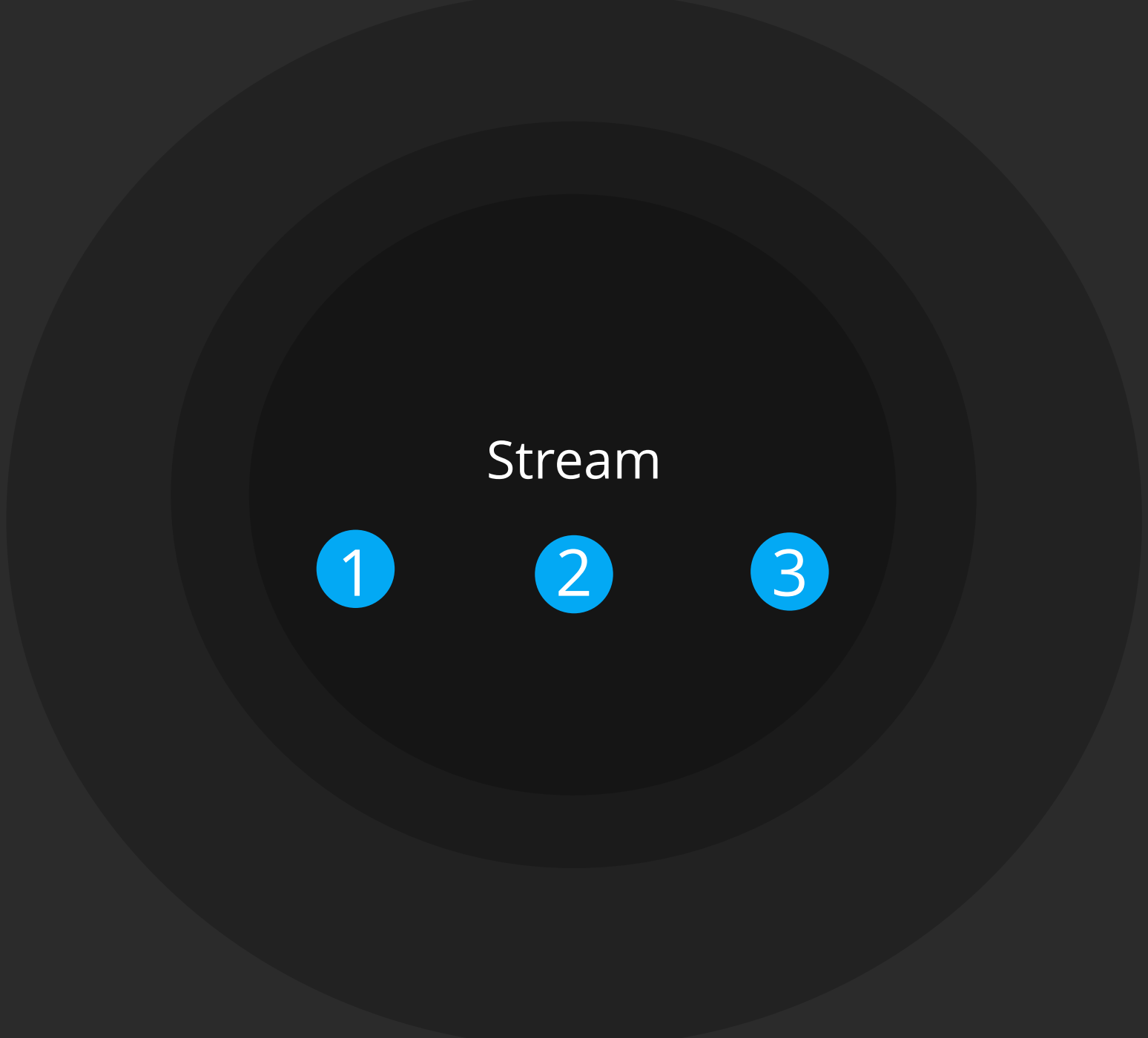
Templating

Template Syntax

Syntax	Binding type
{{title}} <input **[value]**="firstName"> <li **[class.active]**="isActive"> <div **[style.width.px]**="mySize">	Interpolation Property Class Style
<button **(click)**="onClick($event)">	Event
[(ngModel)]="data.value"	Two-way

Reactive Extensions

RxJS 5


Stream
1
2
3

Observable

```
//Observable constructor
let simple$ = Rx.Observable.create(observer => {
  try {
    //pushing values
    observer.next(1);
    observer.next(2);
    observer.next(3);

    //complete stream
    observer.complete();
  }
  catch(e) {
    //error handling
    observer.error(e);
  }
});
```

Subscribe

```
/*
a$ ---1---2---3|
*/

let a$ = Rx.Observable.of(1,2,3);

let subscription = a$.subscribe({
  next:  x => console.log(x),
  error: x => console.log('#'),
  complete: () => console.log('|')
});
```

Unsubscribe

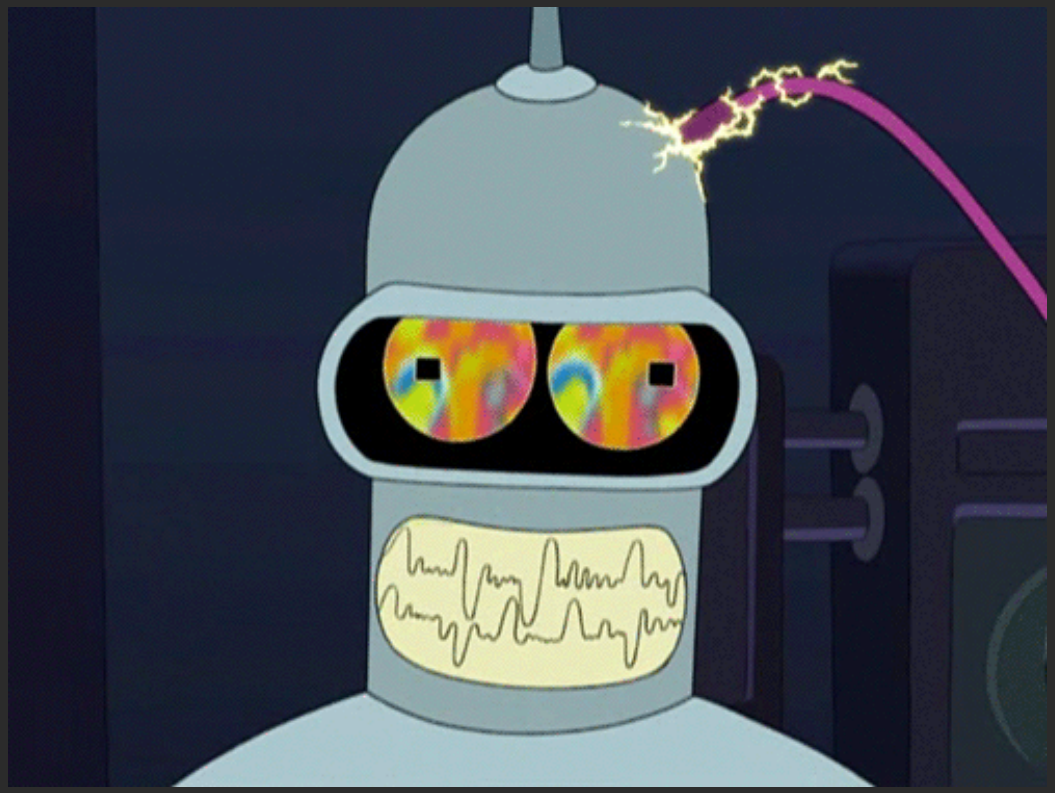
```
let subscription = twits$.subscribe(
  twit => feed.push(twit),
  error => console.log(error),
  () => console.log('done')
);

setTimeout(() => subscription.unsubscribe(), 5000);
```

Example

```
Rx.Observable.of(1)
  .subscribe({
    next: x => console.log(x),
    complete: () => console.log('3')
  });
console.log('2');

// a) 1 2 3
// b) 2 1 3
// c) 1 3 2
// d) 3 2 1
```



Schedulers

```
Observable.of(1)
  .subscribeOn(Rx.Scheduler.async)
  .subscribe({
    next: (x) => console.log(x),
    complete: () => console.log('3')
  });
console.log('2');

// a) 1 2 3
// b) 2 1 3
// c) 1 3 2
// d) 3 2 1
```

Why Observables?

- Flexible: sync or async
- Powerful operators
- Less code

RxJS 5 in Angular2

- Asynchronous processing
- Http
- Forms: controls, validation
- Component events
  - EventEmitter

Http Module

Main Features

- Primary protocol for client/server communications
- Implements XMLHttpRequest (XHR) and JSONP
- Http methods: GET, POST, PUT, DELETE, PATCH and HEAD

Creating a Http Service

```
// app.module.ts
import { HttpModule }  from '@angular/http';
@NgModule({
  imports: [HttpModule], ...
})
export class AppModule {}
```

Creating a Http Service

```
// usersService.ts
import { Injectable } from '@angular/core';
import { Http } from '@angular/http';

@Injectable()
export class UsersService {
  constructor(private http: Http) { }
  get() {
    return this.http.get('/assets/users.json')
      .map(response => response.json().users)
      .retryWhen(errors => errors.delay(2000));
  }
}
```

Consuming a Http Service

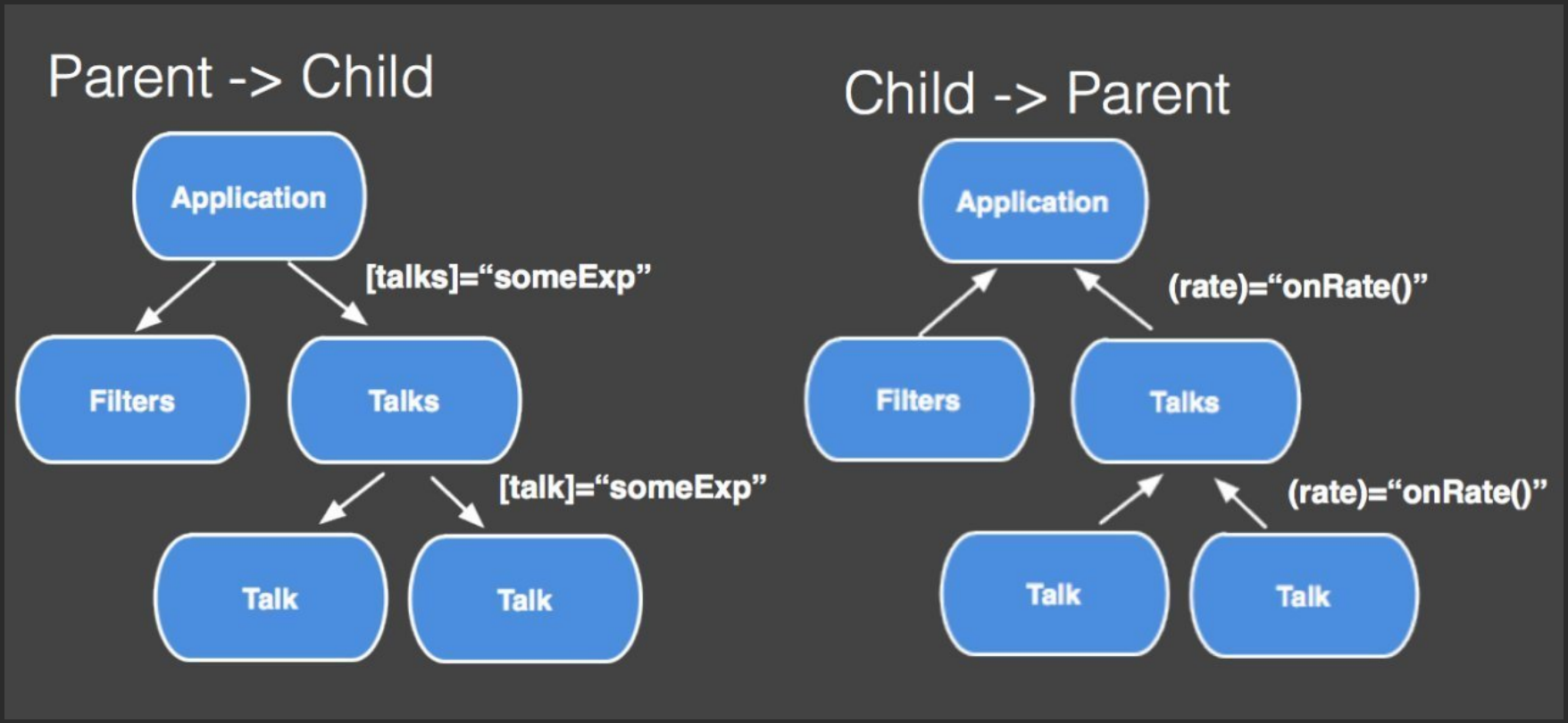
```
import { Component } from '@angular/core';
import { UsersService } from '../services/usersService';

@Component({
  selector: 'users',
  template: `<h1>Users</h1>
    <tr *ngFor="let user of userslist | async">
      <td>{{user.username}}</td>
    </tr>`
})
export class Users {
  private userslist;

  constructor(users: UsersService) {
    this.userslist = users.get();
  }
}
```

Data Architecture

Unidirectional Data Flow



source: blog

Overview

- Data Services Components
- State Management
  - ng2-redux (Redux)
  - ngrx/store (RxJS 5)
- GraphQL/Apollo Client

GDG DevFest
Ukraine 2016

Дякую

Thanks