Angular 2

1.背景介绍

Angular2 作为 Angular 1.x 的后续版本，仍处于测试阶段，但是作为微软和 google 的合作项目，以及当前已经发布出的版本和特性，其未来十分被看好。

那么为什么要打造 Angular2 呢？（此部分摘自 http://www.phonegap100.com/article-453-1.html）

1.性能的限制

AngularJS 当初是提供给设计人员用来快速构建 HTML 表单的一个内部工具。随着时间的推移，各种特性 被加入进去以适应不同场景下的应用开发。然而由于最初的架构限制（比如绑定和模板机制），性能的 提升已经非常困难了。

2.快速变化的 WEB

在语言方面，ECMAScript6 的标准已经完成，这意味着浏览器将很快支持例如模块、类、lambda 表达式、 generator 等新的特性，而这些特性将显著地改变 JavaScript 的开发体验。

在开发模式方面，Web 组件也将很快实现。然而现有的框架，包括 Angular1.x 对 WEB 组件的支持都不够好。

3.移动化

想想 5 年前……现在的计算模式已经发生了显著地变化，到处都是手机和平板。Angular1.x 没有针对移动 应用特别优化，并且缺少一些关键的特性，比如：缓存预编译的视图、触控支持等。

4.简单易用

说实话，Angular1.x 太复杂了，学习曲线太陡峭了，这让人望而生畏。Angular 团队希望在 Angular2 中将复杂性 封装地更好一些，让暴露出来的概念和开发接口更简单。

上述比较笼统得介绍了 Angular2 出现的契机，然后从技术的角度介绍 Angular2 的改变，它吸收了 Angular1 的很多教训+reactjs 的不少优点，具体：（此部分摘自 http://www.zhihu.com/question/26722922/answer/62263149）

- Web Component 组件化
    - 之前也有组件（directive），但是
        - 难写，bug 多
        - 组件之间无法自由组合
    - 2.0 的非常易写，而且类似 jsx， 把 html 和 js 混合，用 decorator 语法（比 jsx 更加人性化）
    - 能使用 polymer 组件
- typescript，es6 的超集，有类型系统，带来的结果是
    - runtime bug 少，在编译阶段解决部分 bug

    - 更好的代码提示
    - 更快的性能

- 单向绑定（不再有 ng-model, $scope，双向绑定等）

    - 性能大升
    - server rendering（这个灰常牛逼，尤其是在你极为注意前端性能的时候， server rendering 可以结合 bigpipe 用， 具体看这里：https://www.facebook.com/notes/facebook-engineering/bigpipe-pipelining-web-pages-for-high-performance/389414033919 要翻墙 中文介绍看这里：BigPipe 学习研究 « 搜索技术博客）
    - 使用简单的语法，可以达到和双向绑定一样的功能

- 什么 module，controller 都没了，只有 es6 的 class

    - 我想说，世界干净多了
    - 从前的 angular， 你很难说清楚什么是你程序中真正的模块

- shadow css

    - css 也变得组件化，而不再是全局规则

- 新的 observe 机制， 不再需要$scope.$apply， 而且检测速度 x5
- etc....

2.搭建基本环境

搭建时参考了官网的 quickstart https://angular.io/docs/ts/latest/quickstart.html
同时也在百度上进行了相关搜索，发现和官网的不太一样，这种随时更新的框架还是得通过官网进行学习。

3.特性 demo
最终 Demo 程序随文档提交。
1) 数据绑定
    - 单向绑定

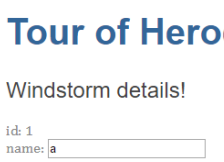
```
@Component({
  selector: 'my-app',
  template: '<h1>{{title}}</h1><h2>{{hero}} details!</h2>'
})
```

此处的双括号告诉我们要去读取 title 和 hero 并且提供数据。

- 双向绑定

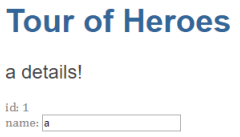
```
<input value="{{hero.name}}" placeholder="name">
```

这样的代码无法使得 hero 改变，此处只是单向绑定。




为了双向绑定，要使用 ngmodel。

```
<input [(ngModel)]="hero.name" placeholder="name">
```




2) 模板数据输入

```
export class AppComponent {
  title = 'Tour of Heroes';
  heroes = HEROES;
  selectedHero: Hero;
  onSelect(hero: Hero) { this.selectedHero = hero; }
}
var HEROES: Hero[] = [
  { "id": 11, "name": "Mr. Nice" },
  { "id": 12, "name": "Narco" },
  { "id": 13, "name": "Bombasto" },
  { "id": 14, "name": "Celeritas" },
  { "id": 15, "name": "Magneta" },
  { "id": 16, "name": "RubberMan" },
  { "id": 17, "name": "Dynama" },
  { "id": 18, "name": "Dr IQ" },
  { "id": 19, "name": "Magma" },
  { "id": 20, "name": "Tornado" }
];
```

这里，heros 是 HERO 的数组。

```
<li *ngFor="let hero of heroes"
```

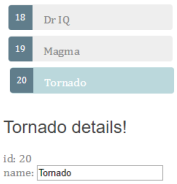
*xxx 这句话的用处就是在该区块中循环 heros 中的 hero 对象。
NgFOR 的（*）前缀意味着 li 元素和它的子元素组成了一个主模板。

效果如下。

3）事件绑定

```
(click)="onSelect(hero)">
```

这里，点击事件将触发 onSelect 事件，并将变量 hero 作为输入。




4）隐藏未被初始化的对象

```
<div *ngIf="selectedHero">
```

当 selectedHero 未被初始化时，ngIf 指令会将其从 dom 中移除直至其被初始化。

以上只是一部分 Angular2 的特性，但足以令我感受到其方便与强大。

4.个人体会

在这之前并未用过框架，因此也无从比较。

将其与原生的 html，css，js 体系比较的话，无疑，使用非常简单，数据交互部分轻松写意，而且网页运行得非常流畅。整个框架主要靠 template，数据填充等撑起，非常的模块化。总之，令我感受到了框架对于开发的重要性。

运行之前需要 npm install　　npm start