Difference between Angular 1.x and Angular

- Angular 1.x was not built with mobile support in mind, where Angular 2 is mobile oriented.

- Angular 2 provides support of TypeScript language.

- Angular 1.x controllers and $scope are gone.We can say that controllers are replaced with "Components" in Angular 2. Angular 2 is component based.

```
var myApp = angular
   .module("myModule", [])
   .controller("productController", function($scope) {
     var prods = { name: "Prod1", quantity: 1 };
     $scope.products = prods;
});
```

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

@Component({
  selector: 'prodsdata',
  template: `
    <h3>{{prods.name}}</h3> `
})

export class ProductComponent {
  prods = {  name: 'Prod1', quantity: 1 };
}
```

- Angular 1.x has 2 ways to bootstrap angular. One using **ng-app** attribute and other via code.

```
<script>
   angular.element(document).ready(function() {
      angular.bootstrap(document, ['myApp']);
});
</script>
```

```
import { bootstrap } from 'angular2/platform/browser';
import { ProductComponent } from './product.component';

bootstrap(ProductComponent);
```

- Structural directives syntax is changed. ng-repeat is replaced with *ngFor.
  Note: local variables are defined using hash(#) prefix

```
<ul>
   <li ng-repeat="technology in technologies">
     {{technology.name}}
   </li>
</ul>
<div ng-if="technologies.length">
   <h3>You have {{technologies.length}} technologies.</h3>
</div>
```

```
<ul>
  <li *ngFor="#technology of technologies">
    {{technology.name}}
  </li>
</ul>
<div *ngIf="technologies.length">
  <h3>You have {{technologies.length}} technologies.</h3>
</div>
```

- To filter output in our templates in Angular 1.x, we use the pipe character (|) and one or more filters. Where in Angular 2 they are called pipes. The syntax remains same.

- Angular 2 uses camelCase syntax for built-in directives. For example, ng-class is now ngClass and ng-model is now ngModel.

- One of the major change in Angular 2 is, that it directly uses the valid HTML DOM element properties and events. Due to this, many of the available built-in directives in Angular 1.x are now no longer required. Like, ng-href, ng-src, ng-show and ng-hide. Angular 2 uses href, src and hidden properties to get the same output. And same goes with event based directives like ng-click and ng-blur.

<button ng-click="doSomething()">	<button (click)="doSomething()">

- In Angular 1.x, ng-bind is used for one way data binding, but with Angular 2 it is replaced with [property], where property is valid HTML DOM element property.

<input ng-bind="technology.name"></input>	<input [value]="technology.name"></input> <div [style.color]="color">Some text...

- In Angular 1.x, ng-model is used for two way data binding, but with Angular 2 it is replaced with [(ngModel)].

<input ng-model="technology.name"></input>	<input [(ngModel)]="technology.name"></input>

- In Angular 1.x, we can define a service via 5 different ways.
    - Factory
    - Service
    - Provider
    - Constant
    - Values

  In Angular 2, class is the only way to define a service.

- One of the advantage of Angular is Dependency Injection. With Angular 2 DI is there but now there is a different way to inject dependencies. As everything is class in Angular, so DI is achieve via constructor.

```
var myApp = angular
   .module("myModule", [])
   .controller("productController", function($scope, $http) {
     var prods = { name: "Prod1", quantity: 1 };
     $scope.products = prods;
});
```

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

@Injectable()
export class TechnologyService {
  constructor(private _http: Http) { }

  getTechnologies() {
    return [
      new technology(1, 'Angular'),
      new technology(2, 'jQuery',
      new technology(3, 'Node'),
      new technology(4, 'Knockout')
    ];
  }
}
```

➢ In Angular 1.x, we use $routeProvider.when() to configuring routing. Where in Angular 2, @RouteConfig{(...}) is used. ng-view present in Angular 1.x is replaced with <router-outlet>

```
var app = angular
       .module("MyModule", ["ngRoute"])
       .config(function ($routeProvider) {
           $routeProvider
             .when("/home", { templateUrl: "home.html", controller: "homeController" })
             .when("/technology", { templateUrl: "technology.html", controller: "technologyController" })
       })
       .controller("homeController", function ($scope) {
           $scope.message = "Home Page";
       })
       .controller("technologyController", function ($scope) {
            $scope.technologies = ["ASP.NET", "jQuery", "AngularJS", "JavaScript"];
       })
```

```
import { Component } from 'angular2/core';
import { RouteConfig, ROUTER_DIRECTIVES, ROUTER_PROVIDERS } from 'angular2/router';
import { TechnologyComponent } from './technology/technology.component';
import { TechnologyService } from './Technology/Technology.service';

@Component({
  selector: 'my-app',
  templateUrl: 'app/app.component.html',
  directives: [ROUTER_DIRECTIVES],
  providers: [
    ROUTER_PROVIDERS,
```

```
    TechnologyService
  ]
})
@RouteConfig([
  { path: '/home', name: 'Home', component: HomeComponent, useAsDefault: true },
  { path: '/technology', name: 'Technology', component: TechnologyComponent },
])
export class AppComponent { }
```

➢ Angular 2 implements webstandards like components and it's provide better performance than Angular 1.

New Topics in Angular2

- Tool Chain, Code Organization
- Asset pipleline hooks
- Programmable CSS with SASS and BourBon
- Transpile ES6 -> ES5
- ES6/ ES7 with Angular
- AMD with Angular
- DI 2.0 with Angular
- HTML5 custom elements
- Functional reactive programming with Angular
- NG-Patterns : Logging, Security, Resiliency, cache