**Key differences between Angular 1 & Angular 2**

**Support for ES6**

Angular 2 is completely written in Typescript and meets the ECMAScript 6 specification. This means that it has support for ES6 Modules, Class frameworks etc.

**Components are new controllers**

In Angular 1 we had Controllers. In Angular 2 Controllers are replaced with Components.The controllers and view in Angular 1 is defined as follows

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11 | //View  <body ng-controller=’appController’>      <h1>vm.message<h1>  </body>  //Controller  angular.module(‘app’).controller(‘appController’,appcontroller)  {      message=’Hello Angular2’;  } |

In angular 2 we are using Components. The simple component is shown below written using Typescript.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12 | import { Component } from '@angular/core';    @Component({      selector: 'app',      template: '<h1>{{message}} </h1>'  })  export class AppComponent  {      message: string=’Hello Angular2’;  } |

In angular 2 a component represents a UI element. You can have many such components in a single web page. Each component is independent of each other and manages a section of the page. The components can have child components and parent components.

**Directives**

The angular1 had a lot of directives. The some of the most used directives are ng-repeat & ng-if

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11 | <ul>      <li ng-repeat =customer in vm.customers>         {{customer.name}}      </li>  </ul>    <div ng-if=”vm.isVIP”>      <h3> VIP Customer </h3>  </div> |

The Angular 2 also has directives, but with a different syntax.It has a \* before the directive name indicating it as a structural directive

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11 | <ul>      <li \*ngFor =#customer of customers>          {{customer.name}}      </li>  </ul>    <div \*ngIf=”vm.isVIP”>      <h3> VIP Customer </h3>  </div> |

The style directives like ng-style, ng-src & ng-href are all gone. These are now replaced by property binding a html elements to the class properties

The creation of Custom Directives is vastly simplified in angular 2 as shown in the example below.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7 | @Directive({      selector: '[MyDirective]'  })  class MyDirective {  } |

**Data Bindings**

The powerful angular data bindings stays the same,  with the minor syntax changes.

**Interpolation**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7 | //Angular 1  <h3> {{vm.customer.Name}}</h3>    //Angular 2  <h3> {{customer.Name}}</h3> |

Note that we used controller alias vm to specify the controller instance in Angular1. In Angular2 the context is implied.

**One way Binding**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7 | //Angular 1  <h3> ng-bind=vm.customer.name></h3>    //Angular 2  <h3 [innerText]=”customer.name” ></h3> |

The angular 2 can bind to any property of the html element.

**Event Binding**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7 | //Angular 1  <button ng-click=”vm.save()”>Save<button>    //Angular 2  <button (click)=”save()”>Save<button> |

The Angular 1 uses the ngClick directive to bind to the event. In Angular 2 ngClick Directive is removed. You can bind directly to the DOM events

**Two- way binding**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7 | //Angular 1  <input ng-model=”vm.customer.name”>    //Angular 2  <input [(ng-model)]=”customer.name”> |

**$scopes are out**

Angular 2 is not using $scope anymore to glue view and controller.

Angular 1 used to run a dirty checking on the scope objects to see if any changes occurred . Then it triggers the watchers. And then it used to re-running the dirty checking again.

The Angular 2 is using zone.js to detect changes. Zone.js apply patches on all the global asynchronous operations like click event, timer events, HTTP requests etc. It then intimates the angular 2 , whenever the changes occur in Angular Application. The Angular 2,then runs the change detection for the entire application

**Filters are renamed to Pipes**

In Angular 1, we used Filters and as shown below

|  |  |
| --- | --- |
| 1  2  3 | <td>{{vn.customer.name | uppercase}}</td> |

Angular 2 uses the same syntax but names them as pipes

|  |  |
| --- | --- |
| 1  2  3 | <td>{{customer.name | uppercase}}</td> |

**Platform specific Bootstrap**

In angular 1 we used ng-app directive in our HTML, then the angular would bootstrap and attach itself the ng-app

|  |  |
| --- | --- |
| 1  2  3  4 | <body ng-app=’app’>  </html> |

The bootstrapping in angular 2 is done through code. The bootstrapping of Angular 2 is not simple as that of Angular 1. The sample code below shows how angular 2 application bootstraps the AppModule using platformBrowserDynamic Module

|  |  |
| --- | --- |
| 1  2  3  4  5  6 | import { platformBrowserDynamic } from '@angular/platform-browser-dynamic';  import {AppModule } from './app.module';    platformBrowserDynamic().bootstrapModule(AppModule); |

The Bootstrap is also Platform specific in angular 2.o. You can have different bootstrapper for mobile & Web application.

You can read about [**Bootstrapping Angular 2 Application**](http://tektutorialshub.com/bootstrapping-angular-2-application/)

**Services**

The Angular 1 had Services, Factories , Providers, Constants and values, which used to create reusable code. These are then injected into Controllers so that it can use it

The angular 2 all the above is merged into a Service.Class.

**Mobile Support**

Angular 1 was not built with mobile support in mind. Angular 2 designed with the mobile development in mind.