VUEJS / ANGULAR COMPARISON

Component	import Component from 'vue-class-component';
	import './app.scss';
	@Component({
	  template: require('./app.html')
	})
	export class AppComponent extends Vue {}
	---------- 	
	new Vue({
	 el: '#app',
	 components: {
	  'app': appComponent
	 }
	});
	=> Or declare child compon. into parent component
		import { Component } from '@angular/core';
		@Component({
		 selector: 'app-root',
		 templateUrl: './app.component.html',
		 styleUrls: ['./app.component.scss']
		})
		export class AppComponent {}
		---------- 	
		import { AppComponent } from 'path/app.component';
		@NgModule({
		 declarations: [
		  AppComponent
		]
		});

Lifecycle hooks	beforeCreate() created() beforeMount() beforeUpdate() updated() beforeDestroy() destroyed()	ngOnChanges() ngOnInit() ngDoCheck() ngAfterContentInit() ngAfterContentChecked() ngAfterViewInit() ngAfterViewChecked() ngOnDestroy()
Text binding	{{ text }}	{{ text }}
HTML binding	v-html=" Text "	[innerHTML]=" Text "
Class binding	:class="{ active: isActive }"	[ngClass]="{'active': isActive}
Link binding	:href="www.test.ch"	[attr.href]="www.test.ch"
Event click binding	@click="function($event)"	(click)="function($event)"
Two-way databinding	v-model="value"	[(ngModel)]="value"

Conditional directive	
	 A
	
	 B
	
	<div v-else>
	 Not A/B/C
	
		<div *ngIf="isValid; then temp1; else temp2">
		  your Content ignored
		
		<ng-template #temp1>
		  your content
		</ng-template>
		<ng-template #temp2>
		  your content
		</ng-template>
		=> Elseif directive doesn't exist

Loop directive	 {{ key }}: {{ value }} 	<div *ngFor="let value of object; let key=index"> {{ key }}: {{ value }}
Input component	<component :xxx="value"></component> import { Prop } from 'vue-property-decorator'; @Props() xxx;	<component [xxx]="value"></component> import { Input } from '@angular/core'; @Input() xxx;

Output component	<component @yyy="func($event)"></component>
	---------- 	
	import { Emit } from 'vue-property-decorator';
	sendData() {
	 this.$emit('yyy', value);
	}
		<component (xxx)="func($event)"></component>
		---------- 	
		import { Output, EventEmitter } from '@angular/core';
		@Output() yyy = new EventEmitter();
		sendData() {
		 this.yyy.emit(value);
		}

Service	***Doesn't exist***	import { Injectable } from '@angular/core';
		@Injectable()
		export class TestService {}
		---------- 	
		import { TestService } from 'path_service/test.service';
		@NgModule({
		  providers: [
		  TestService
		  ]
		});
		=> Use dependency injection to load service

Routing	import VueRouter, { Route, RouteConfig } from 'vue-router';
	Vue.use(VueRouter);
	
	function sendProp(route) {
	  return {propName: route.params.id};
	}
	export const createRoutes: () => RouteConfig[] = () => [{
	  path: '/test/:id',
	  component: testComponent,
	  name: 'test',
	  props: sendProp(),
	  beforeEnter: (to, from, next) => funcGuard(next)
	}];
	---------- 	
	new Vue({
	  router: createRouter(),
	});
		export const createRoutes: Routes = [{
		  path: '/test/:id',
		  component: testComponent,
		  canActivate: [AuthguardService]
		}];
		---------- 	
		import { RouterModule } from '@angular/router';
		@NgModule({
		  imports: [
		  RouterModule.forRoot(createRoutes, { useHash: false })
		  ]
		});

Link Route from template	<router-link :to="{ name: 'test', params: { id: 123 }}"
	  tag="a"
	  active-class="active"
	  exact>Home</router-link>
		<a routerLink="/test/123"
		  routerLinkActive="active"
		  [routerLinkActiveOptions]="{exact: true}">Home

Link Route from component	import { Router } from '@angular/router';
	@Component({})
	export class TestComponent {
	  constructor(private router: Router) { }
	  gotoTo() {
	  this.$router.push({ name: 'test', params: { id: 123 }});
	  }
	}
		import { Router } from '@angular/router';
		@Component({})
		export class TestComponent {
		  constructor(private router: Router) { }
		  gotoTo() {
		  this.router.navigate(['/test/123']);
		  }
		}

Fetch Route parameter	***Using Prop()***	import { ActivatedRoute } from '@angular/router';
		@Component({})
		export class TestComponent implements OnInit {
		  constructor(private route: ActivatedRoute) { }
		  ngOnInit() {
		  this.route.params.subscribe(params => {});
		  }
		}

HTTP Calls	import axios, {AxiosResponse} from 'axios';
	
	axios.get(endpoint)
	  .then((response) => {console.log(response.data)});
	  .catch((error) => {});
	
	axios.post(endpoint, object)
	  .then((response) => {console.log(response.data)});
	  .catch((error) => {});
	=> Axios return a promise
		import { HttpClient } from '@angular/common/http';
		
		this.http.get(endpoint)
		  .map((response) => {return response});
		  .catch((error) => {});
		
		this.http.post(endpoint, object)
		  .catch((error) => {});
		=> HttpClient uses dependency injection (service)
		=> It will return an observable (use subscribe)