Angular Fundemental & Best Practices

# Template Variable

* '#' - variable declaration
* '()' - event binding
* '[]' - property binding
* '{{ }}' - interpolation
* '\*' - structural directives

## variable declaration

used to access variable between HTML and Typescript, like data-dojo-attach-point

### access DOM

template:

|  |
| --- |
| <div #dataContainer></div> |

component:

|  |
| --- |
| import { Component, ViewChild, ElementRef, Renderer2 } from '@angular/core';  @Component({  templateUrl: "some html file"  })  export class MainPageComponent {  @ViewChild('dataContainer') dataContainer: ElementRef;  constructor(private \_renderer: Renderer2) {  this.dataContainer.nativeElement.innerHTML = 'hello';  //or  this.\_renderer.setProperty(this.dataContainer.nativeElement, 'innerHTML', 'hello');  }  } |

### access single component

template:

|  |
| --- |
| <user-profile #myProfile (click)="update()"></user-profile> |

component:

|  |
| --- |
| export class MasterPage {  @ViewChild('myProfile') userProfile: UserProfile  constructor() { }  update(){  this.userProfile.sendData();  }  } |

### ccess multiple component

template:

|  |
| --- |
| <div \*ngFor="let v of views">  <customcomponent #cmp></customcomponent>  </div> |

component:

|  |
| --- |
| /\*\* Get handle on cmp tags in the template \*/  @ViewChildren('cmp') components:QueryList<CustomComponent>;  ngAfterViewInit(){  // print array of CustomComponent objects  console.log(this.components.\_results);  } |

## event and property binding

template:

|  |
| --- |
| <div class="app">  Parent: {{ myCount }}  <counter  bind property, mycount is variable in class below  [count]="myCount"  bind event  (change)="countChange($event)">  </counter>  </div>  export class AppComponent {  myCount: number = 10;  countChange(event) {  ...  }  } |

component:

|  |
| --- |
| import { Component, Input, Output } from '@angular/core';  @Component({...})  export class CounterComponent {  @Input()  count: number = 0;  @Output()  change: EventEmitter<number> = new EventEmitter<number>();  increment() {  this.count++;  this.change.emit(this.count);  }  decrement() {  this.count--;  this.change.emit(this.count);  }  } |

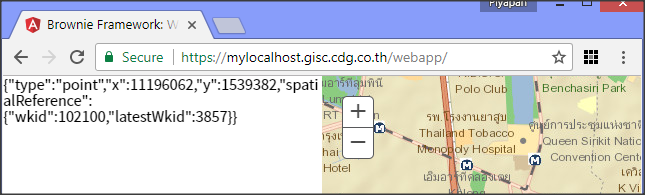
## interpolation

template:

|  |
| --- |
| <div> {{extentInfo}} </div> |

component:

|  |
| --- |
| import {Component, OnInit, NgZone} from '@angular/core';  import {GlobalVariables} from '../global';  import \* as esriLoader from 'esri-loader';  @Component({selector: 'my-content-pane', templateUrl: './myContentPane.component.html', styleUrls: ['./myContentPane.component.scss']})  export class MyContentPane implements OnInit  {  extentInfo: string = 'Welcome';  constructor(private ngZone: NgZone)  {  }  ngOnInit()  {  //if you have to update variable in callback function, put it inside ngZone  GlobalVariables.onMapLoad((map) => this.ngZone.run(() =>  {  this.extentInfo = JSON.stringify(map.extent.getCenter());  }));  }  } |



## structural directives

<https://angular.io/guide/structural-directives>

put logic in HTML

|  |
| --- |
| //check if hero != null  <div \*ngIf="hero" >{{hero.name}}</div>  <ul>  <li \*ngFor="let hero of heroes">{{hero.name}}</li>  </ul>  <div [ngSwitch]="hero?.emotion">  <happy-hero \*ngSwitchCase="'happy'" [hero]="hero"></happy-hero>  <sad-hero \*ngSwitchCase="'sad'" [hero]="hero"></sad-hero>  <confused-hero \*ngSwitchCase="'confused'" [hero]="hero"></confused-hero>  <unknown-hero \*ngSwitchDefault [hero]="hero"></unknown-hero>  </div> |

# ngModel

<https://stackoverflow.com/questions/43234559/what-is-the-difference-between-the-name-and-ngmodel-name-in-angular2-form>

<input [(ngModel)]="foo">

is the short form of

[ngModel]="foo" (ngModelChange)="foo = $event"

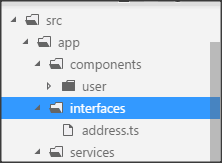
which shows it is for two-way binding.

# Angular Components

* Interface used to define JSON structure.
* Class is interface with constructor (default value)
* Component is like widget in dojo
* Service is global variable
* Directive is used to change css/behavior of DOM.

## Create using CLI

<https://github.com/angular/angular-cli>

e

create folder components, interfaces, classes and services under app folder first

then, type command

|  |
| --- |
| ng g class classes/<name>  ng g interface interfaces/<name>  ng g component components/<name>  ng g service services/<name> |

## Class vs Interface

<https://stackoverflow.com/questions/37652801/when-to-use-interface-and-model-in-typescript-angular2>

Interface

|  |
| --- |
| export interface Product {  ProductNumber: number;  ProductName: string;  ProductDescription: string;  } |

Class

|  |
| --- |
| export class Product {  constructor(  public ProductNumber: number,  public ProductName: string,  public ProductDescription: string  ){}  } |

Interface **is NOT compiled to javascript since it does not exist in javascript**. So, interface acts like type checking to prevent bug when coding.

|  |
| --- |
| this.http.get('...')  .map(res => <Product[]>res.json()); |

Class will compile to javascript, so you have to do like this

|  |
| --- |
| this.http.get('...')  .map(res => {  var data = res.json();  return data.map(d => {  return new Product(d.productNumber,  d.productName, d.productDescription);  });  }); |

## Directives

<https://angular.io/guide/structural-directives>

|  |
| --- |
| import { Directive, ElementRef, Input } from '@angular/core';  @Directive({ selector: '[myHighlight]' })  export class HighlightDirective {  constructor(el: ElementRef) {  el.nativeElement.style.backgroundColor = 'yellow';  }  } |

and add to declarations in NgModule

|  |
| --- |
| @NgModule({  imports: [ BrowserModule ],  declarations: [  AppComponent,  HighlightDirective  ],  bootstrap: [ AppComponent ]  })  export class AppModule { } |

Then you can use like this

|  |
| --- |
| <p myHighlight>Highlight me!</p> |

## Pipes

<https://angular.io/guide/pipes>

|  |
| --- |
| import { Pipe, PipeTransform } from '@angular/core';  /\*  \* Raise the value exponentially  \* Takes an exponent argument that defaults to 1.  \* Usage:  \* value | exponentialStrength:exponent  \* Example:  \* {{ 2 | exponentialStrength:10 }}  \* formats to: 1024  \*/  @Pipe({name: 'exponentialStrength'})  export class ExponentialStrengthPipe implements PipeTransform {  transform(value: number, exponent: string): number {  let exp = parseFloat(exponent);  return Math.pow(value, isNaN(exp) ? 1 : exp);  }  } |

then you can used in HTML like this

|  |
| --- |
| <p>Super power boost: {{2 | exponentialStrength: 10}}</p> |

## Guard

used to protect angular routes

## Enum

<https://www.gurustop.net/blog/2016/05/24/how-to-use-typescript-enum-with-angular2/>

**Not recommended** because it can confuse if you use it wrong.

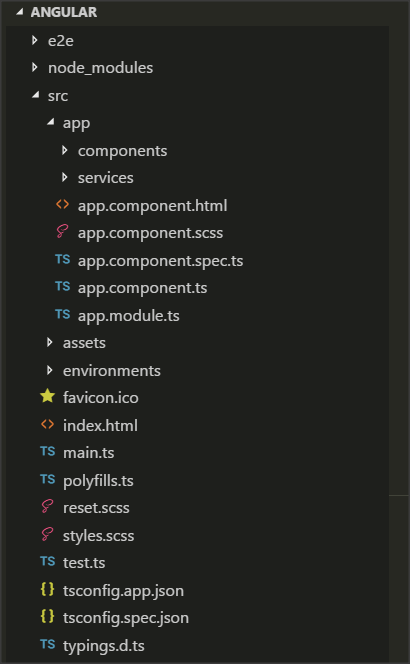
# Create Component Programmatically

assume that component name is infoForm

|  |
| --- |
| import { Component, Injector, ApplicationRef, ComponentFactoryResolver, ElementRef, ComponentRef } from '@angular/core';  import { infoForm } from '../../widgets/infoForm/infoForm';  @Component({  selector: 'argis-lab',  templateUrl: 'lab.html'  })  export class LabPage {  widgetRef: ComponentRef<infoForm>;  constructor(  private injector: Injector,  private componentFactoryResolver: ComponentFactoryResolver,  private appRef: ApplicationRef)  {  //create div  var myDiv = document.createElement('div');  let factory = this.componentFactoryResolver.resolveComponentFactory(infoForm);  //destroy old widget to prevent memory leak  if(this.widgetRef != null)  {  this.appRef.detachView(this.widgetRef.hostView);  //remove event listener  this.infoWidgetListenFunc.unsubscribe();  }  //create widget programmatically  this.widgetRef = factory.create(this.injector, null, myDiv);  //set parameter  this.widgetRef.instance.locationName = clickedGraphic.attributes.LocationName;  this.appRef.attachView(this.widgetRef.hostView);  //listen events  this.infoWidgetListenFunc = this.infoWidgetRef.instance['my\_zoom\_event'].subscribe(ev => {  ...  });  }  } |

# Project Structure

## FrontEnd Structure

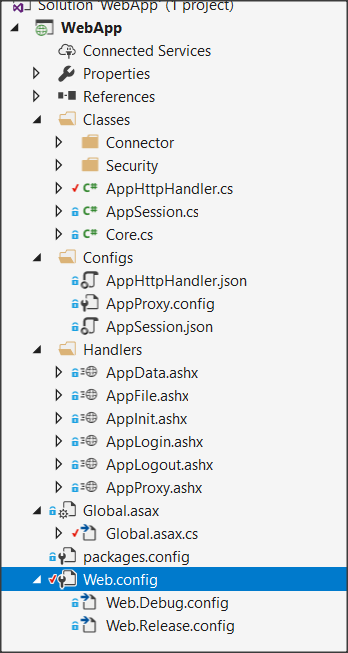


* packages.json เป็นไฟล์ของ node ที่ใช้ในการ Build project
  + zone.js version >= 0.8.13 ตอนนี้มีปัญหากับ IE11 ที่ทำให้ pan แผนที่ไม่ได้ ดังนั้นตอนนี้เลย fix ให้เป็น version 0.8.12 ก่อน
* styles.scss เป็นไฟล์ global css
* reset.scss เป็นไฟล์ที่อุ้ยอวบเพิ่มขึ้นมา เพื่อทำให้ default css ของทุก browser เหมือนกัน
* polyfills.ts เป็นไฟล์ที่เสริม function ใน browser เก่าให้ทำงานได้ เช่นตอนนี้เราเสริม function javascript ใน IE10 ให้ทำงาน angular ได้อยู่ (เราไม่ทำไปถึง IE9 เพราะตอนนี้มี bug อยู่)
* index.html เป็นไฟล์แรกที่ถูกเรียก โดยเราสามารถใส่ global css ลงไปในไฟล์นี้ได้ เช่นตอนนี้ที่มีการ load css ของ arcgis เพิ่มลงไป
* app/components ไว้เก็บ component
* app/services ไว้เก็บ services

สิ่งที่ต้องแก้เพิ่มเติม

* แก้การ load arcgis api ให้เป็นแบบ angular ซึ่งตอนนี้ทำผิดอยู่ (ต้องทำตาม <https://github.com/tomwayson/angular-esri-loader>)
* ที่เหลือเขียนไว้ใน <https://gitlab.gisc.cdg.co.th/Brownie.Framework/Web/WebApp/issues>

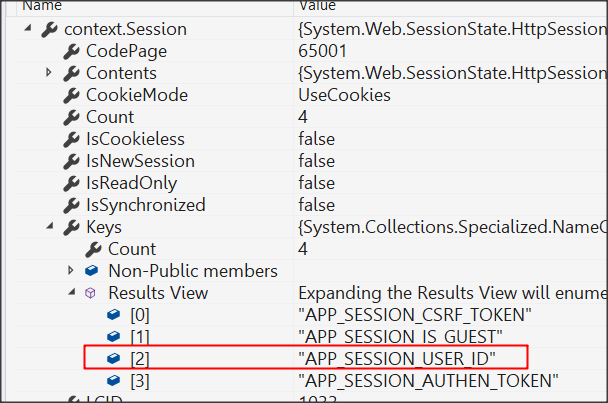
## BackEnd Structure



* Classes ไว้เก็บไฟล์ c# ทั้งหมด
* Configs
  + AppHttpHandler.json จะมาแทนที่ web.config โดยเป็นไฟล์แบบ json เขียนได้ง่ายขึ้น
  + AppSession.json จะประกอบไปด้วย
    - SessionItems : ไว้บอกว่าตัวแปร Session เรามีตัวแปรชื่ออะไรบ้าง
    - SessionItemPrefix : ไว้เป็นคำนำหน้าในชื่อของตัวแปร SessionItems เพื่อป้องกันชื่อตัวแปรซ้ำกับตัวแปรตั้งต้นของ Session

เช่น

|  |
| --- |
| "SessionItemPrefix": "APP\_SESSION\_",  "SessionItems": [  {  "Name": "USER\_ID"  "Value" : <ค่าเริ่มต้นที่ต้องการ>  }  ] |



เวลาต้องการจะ set, get ก็ใช้ method แบบนี้

|  |
| --- |
| AppSession.SetSession("IS\_GUEST", false)  AppSession.GetSession("IS\_GUEST") |

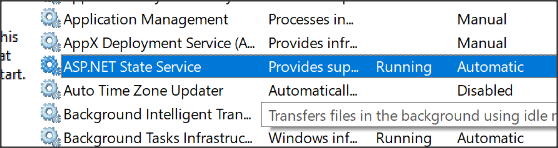
* + ProcedureSessionParameters เป็นสิ่งที่พี่อุ้ยอวบเพิ่มขึ้นมา โดยจะส่งค่า Session ที่ระบุเข้า Store มาเรียกผ่าน AppData.ashx (AppData.ashx คือ DS\_TRANSIT)
* ashx
  + AppData.ashx จะมาแทนที่ DS\_TRANSIT
  + AppFile.ashx ไว้จัดการการ upload ไฟล์
  + AppInit.ashx ไว้ให้ web เรียกครั้งแรกเพื่อต้องการ init ค่า เช่น เมื่อต้องการให้ web ส่งค่าใน web.config กลับมาตอนเรียกใช้ครั้งแรก
  + AppLogin.ashx
  + AppLogout.ashx
  + AppProxy.ashx ก็คือ Proxy.ashx นั้นแหละ

### web.config

เราให้ sessionState mode="StateServer" เพราะต้องการให้ session คงอยู่แม้ว่าเราจะมีการ kill process บน IIS ก็ตาม

ดังนั้น ถึงแม้ว่าเราจะ update code ผู้ใช้ก็ยัง login อยู่นะ

เราจะต้องไปเปิด service ใน Windows ด้วย



# Security

## CSRF

abc

# Database Connection

## sql server

