**คอมโพเน้นท์**

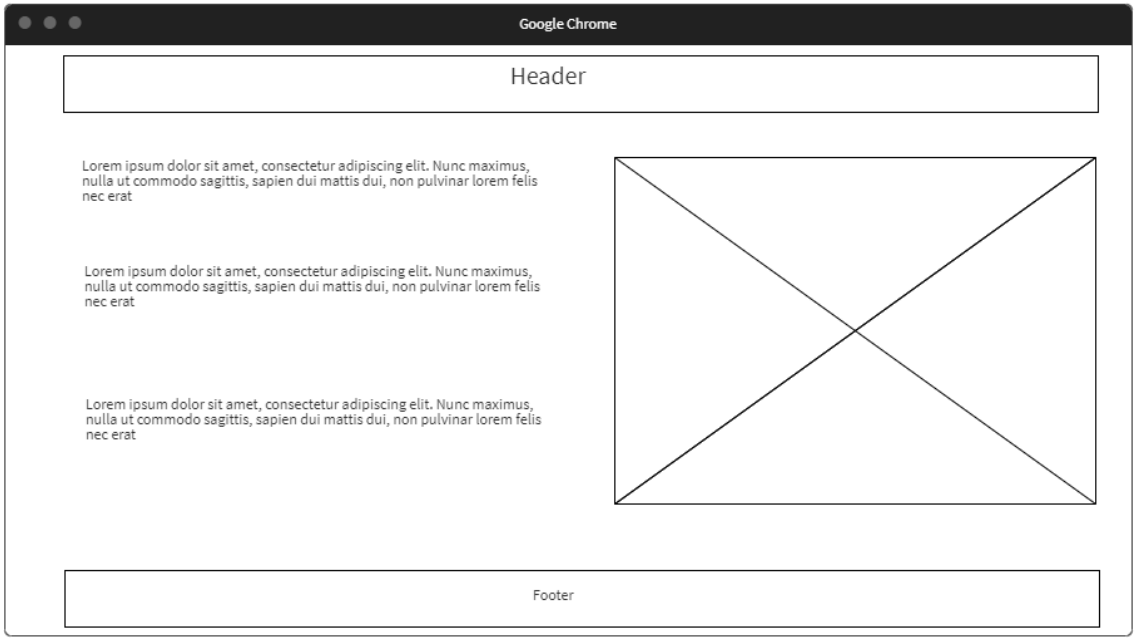
ในบทเรียนนี้ เราจะมาเริ่มใช้คอมโพเน้นท์ ซึ่งเป็นส่วนประกอบหลักที่ใช้แสดงผลหน้าเว็บ ในคอมโพเน้นท์หนึ่งจะประกอบด้วยไฟล์สำคัญสามชนิด คือ TS, HTML, และ CSS แต่ละไฟล์แยกการทำหน้าที่ กล่าวคือ ไฟล์ ประเภท TS ทำหน้าควบคุมการทำงาน และจัดการการแสดงผล ไฟล์ HTMS, CSS ทำหน้าที่แสดงผล หากมองในมุมมอง MVC (Model-View-Controller) หรือ MVVM (Model-View-ViewModel) ไฟล์ TS ทำหน้าที่ Controller/ViewModel และ HTML/CSS แทน View ซึ่งเราจะทดลองใช้ทั้งสามไฟล์นี้ เพื่อใช้เป็นโลจิกและแสดงผล ดังนั้นในบทนี้เราจะทำความเข้าใจในเรื่องต่อไปนี้

- การสฎ่างเทมเฟลท (Template)
- การสร้างโลจิกใน HTML: ngFor, ngIf, ngSwitch
- การสร้างไดเร็กทีฟ(Directive) และไปป์ (Pipe)
- การใช้งาน CSS, Bootstrap, JQuery

---

**สมมุติงานเว็บ**

ก่อนที่จะเรียนรู้การใช้ Angular ต่อไป เรามาสมมุติงานที่จะสร้างเว็บกีก่อน โดยคิดว่าจะสร้างเว็บ หน้าแรกเป็น หน้าเว็บแนะนำข้อมูลทั่วไป เช่น ประกาศข่าว และมีเมนูต่างๆ คือ Home, Courses, และ About Me โดยหน้า Home ก็คือหน้าแรกนี้ เป็นรายการแสดงรายวิชา ที่มีให้เลือกเรียนกัน ซึ่งมีประมาณเริ่มต้นที่ 3 วิชา และหน้า About Me เป็นการแนะนำข้อมูลเบื้องต้นเดียวกับเว็บนี้ โครงสร้างหน้าเว็บ ผลตามรู้ต่อไปนี้




**รูป 1** หน้าแรก (สร้างจาก https://wireframepro.mockflow.com)

**template / templateUrl**

คอมโพเน้นท์เริ่มต้นใช้ ตามค่าปริยาย ที่ใช้สร้างก่อนหน้านี้ ที่ไม่ได้แก้ไขอะไร ให้เป็นไปตามที่ Angular สร้างมาให้จากเดิม ไฟล์ app.component.ts จะมีลักษณะดังต่อไปนี้

**Code 1.** src/app/app.component.ts

```
import { Component } from '@angular/core';

@Component({
  selector: 'app-root',
  templateUrl: './app.component.html',
  styleUrls: ['./app.component.css']
})
export class AppComponent {
  title = 'myAngular';
}
```

หากจำกันได้ ไฟล์นี้ ใช้สำหรับแสดงผล โดยมี แม่แบบ (templateUrl) จากไฟล์ app.component.html และกำหนดรูปแบบ จากไฟล์ app.component.css การนำไปแสดงผลในตำแหน่งอิลีเม้นท์ app-root ที่จะปรากฏในไฟล์ index.html

การแสดงผล นอกจากจะสร้าง จากไฟล์แม่แบบ ยังสามารถสร้างได้ในไฟล์ app.component.ts เอง ซึ่งการสร้างในไฟล์ตัวเอง จะใช้การกำหนดเป็น template แทน templateUrl เราลองมาสร้างในไฟล์ตัวเอง โดยดัดแปลง ใหม่ ดังนี้

**Code 2.** src/app/app.component.ts

```
import { Component } from '@angular/core';

@Component({
  selector: 'app-root',
  template: `
             <h1>Started Computer Learning at SBC</h1>
             `,
  styleUrls: ['./app.component.css']
})
export class AppComponent {
  title = 'myAngular';
}
```

ตอนนี้เราสร้างเพียง <h1></h1> ให้สังเกตว่า เราใช้เครื่องหมาย ` (back tick) สัญลักษณ์นี้อยู่มุมบนสุดซ้ายสุดของคีย์บอร์ด โดยใช้แทนเครื่องหมายคำพูด เครื่องหมายนี้ใช้เพื่อระบุว่า ข้อความภายในเครื่องหมายนี้เป็น ข้อความ HTML และสัญลักษณ์นี้เป็นมาตรฐานของ ภาษา JavaScript รุ่น ECMAScript 2015 เป็นต้นไป

แต่ดูเหมือนว่า หากต้องเขียน HTML จำนวนมาก การใช้ template คงไม่สะดวก เพราะไม่สามารถแยกงานให้เป็นอิสระกว่าการใช้ templateUrl จึงควรกลับมาใช้แบบ templateUrl แบบเดิมน่าจะดีกว่า (ใช้ตาม **Code 1**)

ตามค่าปริยายแล้ว การสร้างคอมโพเน้นท์จะมีเทมเพลทไฟล์มาให้ หากเราไม่ต้องการให้สร้างเทมเพลทไฟล์ ให้สร้างด้วยคำสั่ง:

```
ng generate component component_name -it
```

เมื่อสร้างคอมโพเน้นท์ใดขึ้นมา จะมีการเพิ่มคอมโพเน้นท์ลงในไฟล์ app.module.ts ให้อัตโนมัติ การเพิ่มจะเพิ่มในส่วน import และ ในส่วน declarations

**กำหนดค่าเริ่มต้นผ่านคอนสตรักเตอร์**

ในการกำหนดค่าเริ่มต้น ที่ต้องการใช้กับคอมโพเน้นท์ สามารถกำหนดค่าได้ที่ตัวแปรของในคลาส (ตัวแปรของออบเจ็กต์) อย่างการกำหนดค่า title = 'myAngular' ได้ แต่ยังมีอีกวิธีหนึ่งที่กำหนดค่าเริ่มต้นได้ผ่านคอนสตรักเตอร์ (คลาสทุกคลาสสามารถกำหนดคอนสตรักเตอร์ได้) ซึ่งจะได้ค่าที่ใช้สำหรับแต่ละออบเจ็กต์ (คลาสต้องสร้างเป็นออบเจ็กต์ก่อนจึงจะนำใช้งานได้)

จากตัวอย่างงานเว็บสมมุติ ให้ มีรายวิชาที่จะแจ้งเปิดสอน ซึ่งเก็บข้อมูลข่าวในรูปอาร์เรย์ ดังนั้นในที่นี้เราจะสร้างรายการข่าว ที่มีค่าเริ่มต้นในคอนสตรักเตอร์ ของคลาส AppComponent

**Code 3.** src/app/app.component.ts

```
import { Component } from '@angular/core';

@Component({
  selector: 'app-root',
  templateUrl: './app.component.html',
  styleUrls: ['./app.component.css']
})
export class AppComponent {
  title = 'myAngular';
  news = new Array();
  constructor(){
        this.news.push({title:'C# Programing',
                        content:'C# Programming will start on 12 Jan. 2019',
                        active:true, important:1, expire:'2018/4/30'});
        this.news.push({title:'JS Programing',
                        content:'JS Programming will start on 12 Feb. 2019',
                        active:true, important:2, expire:'2018/4/30'});
        this.news.push({title:'Python Programing',
                        content:'R Programming will start on 12 Apr. 2019',
                        active:false, important:3, expire:'2018/5/30'});
  }
}
```

ในตัวอย่างนี้ ใช้อาร์เรย์แบบ ทูเพิล (tuple) สามารถเพิ่มออบเจ็กต์เป็น JSON ด้วยฟังก์ชัน push( ) ได้ ซึ่งได้เพิ่มเข้าไป สามข่าว ดังนั้นเมื่อคอมโพเน้นท์นี้ทำงาน จะมีสามข่าวนี้อยู่ในคอมโพเน้นท์นี้ด้วย

**สร้างโลจิกใน HTML**

ในภาษาเขียนโปรแกรม มีโลจิกที่สำคัญคือ if..else, for.. และ switch..case เราสามารถใส่โลจิกเหล่านี้ได้ในเอกสาร HTML ผ่านคำสั่ง ไดเร็กทีฟ (directives) ซึ่งจะใช้ เครื่องหมาย * นำหน้า หรือใช้ เครื่องหมาย [ ]

*ngIf แทนเงื่อนไข "ถ้า" ซึ่งถ้าเงื่อนไข เป็นจริง (true) ก็จะแสดงผล ในส่วน  ที่มี *ngIf ประกาศในนี้ และถ้าเงื่อนไขเป็นเท็จ จะทำงานในส่วนของ else ก็จะแสดงผลใน <ng-template #emptyNews> กรณีที่ไม่ต้องการใช้ else ก็ละส่วนการเพิ่ม else ไปได้เลย และการเพิ่มเงื่อนไข OR และ AND จะแทนด้วยเครื่องหมาย || และ && ดังตัวอย่างการใช้ต่อไปนี้

```
<div *ngIf='Expression'></div>
<div *ngIf='Expression1 || Expression2'></div>
<div *ngIf='Expression1 && Expression2'></div>
```

หรือจะสร้างเป็น ng-template ทั้งในส่วน ngIf และ else จะมีรูปแบบการใช้คือ:

```
<ng-container *ngIf='Expression; then templateYes; else templateNo'>
</ng-container >
<ng-template #templateYes>Yes</ng-template>
<ng-template #templateNo>No</ng-template>
```

*ngFor ใช้ในการวนซ้ำ ซึ่งใช้คู่กับ of เช่น ตามรูปแบบต่อไปนี้มี item เป็นตัวแปรในการวนซ้ำ และมี collection เป็นชุดข้อมูล การวนซ้ำจะทำงานภายใต้  ที่ *ngFor อาศัยอยู่

```
<div *ngFor="let item of collection>item</div>
```

ยังมีอีกโลจิก คือ *ngSwitch ซึ่งเป็นการเลือกทำตามตัวเลือก มีรูปแบบการใช้งานดังนี้

```
<div [ngSwitch]='Expression'></div>
<div *ngSwitchCase="'case1'"></div>
<div *ngSwitchCase="'case2'"></div>
<div *ngSwitchCase="'case3'"></div>
<div *ngSwitchDefault></div>
```

ต่อไปจะนำชุดข้อมูลข่าวแสดงที่หน้าเว็บ ซึ่งอยู่ที่ไฟล์ app.component.html ตามที่ระบุใน templateUrl และแสดงตัวอย่างการใช้โลจิกต่าง ๆ ทั้งใช้การวนซ้ำในอาร์เรย์ การตั้งเงื่อนไข ถ้า..แล้ว และ เลือกทำตามตัวเลือก ดังการปรับปรุงไฟล์ในเทมเพลทคือ

**Code 4.** src/app/app.component.html

```
<div >
 <header>
   <h1>Started Learning Information Technology</h1>
 </header>
 <div>
       <div>
              <div *ngFor="let item of news">
                {{item.content}}
                <span [ngSwitch]='item.important'>
                       <span *ngSwitchCase=1>สำคัญมาก</span>
                       <span *ngSwitchCase=2>สำคัญ</span>
                       <span *ngSwitchDefault></span>
                </span>
              </div>
              <div *ngIf='news.length>0 else emptyNews'>
                     มีข่าวแจ้ง({{news.length}})
              </div>
              <ng-template #emptyNews>
```

```
                ขณะนี้ไม่มีข่าว
            </ng-template>
        </div>
        <div>
            <img src="./assets/images/sbc_building.jpg">
    </div>
 </div>
 <footer>
        <p>Information Technology Department</p>
 </footer>
</div>
```

จากตัวอย่างนี้ ใช้ *ngFor เป็นตัววนซ้ำในอาร์เรย์ โดยมีตัววิ่งในอาร์เรย์ซึ่งแทนข้อมูลข่าวแต่ละรายการในชื่อ item และทำการผูกข้อมูลด้วยเครื่องหมายปีกกาคู่ การใช้ [ngSwitch] จับค่า important ซึ่งเป็นค่าตัวเลข 1,2,3 โดยให้ค่า 3 เป็นตัวเลือกปริยาย (*ngSwtchDefault) ให้สังเกตว่า เฉพาะตัว [ngSwitch] ใช้เครื่องหมาย [ ] ส่วนตัวอื่น ๆ ใช้ เครื่องหมาย * สำหรับ *ngIf .. else ใช้เทมเพลท สำหรับ else

ถึงตอนนี้เมื่อทำการบันทึกข้อมูล หน้าเว็บจะเปลี่ยนแปลง โดยแสดงของมูลเป็นรายการทั้งสามข่าว (ดูที่เบราเซอร์) มีหัวข้อ (header) มีข่าว กับรูปภาพ (content) และมีข้อมูลท้ายเว็บ (footer) ตามอีลีเม้นท์  สำหรับรูปภาพต้องนำไฟล์รูปไปใส่ตามเส้นทางโฟลเดอร์ที่ระบบใน src/assets/images

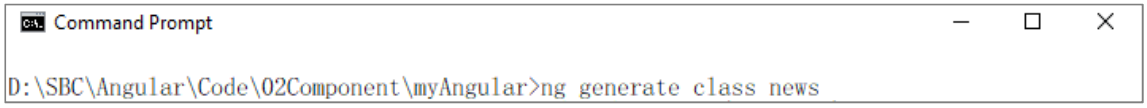

**รูป 2** ผลแสดงหน้าเว็บ

**สร้างคลาสเพื่อเก็บโครงสร้างข้อมูล**

ที่ผ่านมาเราสร้างข้อมูลข่าว ผ่านคอนสตรักเตอร์ได้ แต่จะเป็นการดีที่จะแยกข้อมูลให้เป็นอิสระเพื่อการเรียกใช้ซ้ำในคอมโพเน้นท์อื่นๆ โดยการสร้างไฟล์หรือคลาสสำหรับเก็บข่าวโดยเฉพาะ เราดำเนินการโดย ใช้ คำสั่งตามรูปแบบต่อไปนี้

```
ng generate class class_name
```

ในที่นี้เราต้องการสร้างคลาส News ให้ เวลาพิมพ์ใส่เป็นอักษรพิมพ์เล็กได้แต่เมื่อสร้างเป็นคลาสแล้ว จะได้ชื่อคลาสเป็นอักษรพิมพ์ใหญ่โดยอัตโนมัติ ดังตัวอย่างการสร้างคลาส News ที่มีเส้นทางเริ่มต้นของไฟล์เว็บ ณ ตำแหน่ง myAngular ตามรูปต่อไปนี้




**รูป 3** การสร้างคลาส News

เมื่อสร้างคลาส News ได้แล้ว จะได้โครงของคลาส ซึ่งเราต้องสร้างสมาชิกภายในคลาสเอง สมาชิกที่จะสร้างให้กับคลาส เราจะให้มีสมาชิกตาม โครงสร้างเดิม คือ มี title, content, active, และ expire ดังนี้แล้ว คลาสที่สร้างใหม่นี้จึงมีลักษณะดังนี้

**Code 5.** src/app/app.news.ts

```
export class News {
      constructor(
            public title:string,
            public content:string,
            public active:boolean,
            public important:number,
            public expire:string){    }
}
```

คลาสที่สร้างนี้ สร้างสมาชิกผ่านคอนสตรักเตอร์ ได้เลย ซึ่งเป็นรูปแบบการสร้างแบบใหม่ และดูต่างจากภาษาเชิงวัตถุอื่นๆ มาก

เมื่อได้คลาสที่มีสมาชิกแล้ว ต่อไปจะกำหนดค่าของสมาชิกต่างๆ ภายในคอมโพเน้นท์หลัก ขั้นตอนแรกคือต้องนำเข้าคลาส News ก่อน แล้วสร้างออบเจ็กต์ News พร้อมๆ กับใส่ข้อมูลลงในออบเจ็กต์ ดังตัวอย่างต่อไปนี้

**Code 6.** src/app/app.component.ts

```
import { Component } from '@angular/core';
import { News } from './news';                //add this line

@Component({
  selector: 'app-root',
  templateUrl: './app.component.html',
  styleUrls: ['./app.component.css']
})
export class AppComponent {
  title = 'myAngular';
  news = new Array();
  constructor(){
      //change these following lines
      this.news.push(new News('C# Programing',
                              'C# Programming will start on 12 Jan. 2019',
                              true, 1, '2018/4/30'));
```

```
      this.news.push(new News('JS Programing',
                              'JS Programming will start on 12 Feb. 2019',
                              true, 2, '2018/4/30'));
      this.news.push(new News('R Programing',
                              'R Programming will start on 12 Apr. 2019',
                              false, 3, '2018/5/30'));
  }
}
```

ถึงตอนนี้เราได้หน้าเว็บ ที่มีข้อมูลตามต้องการเรียบร้อยแล้ว แต่ยังขาดส่วนการกำหนดรูปแบบ CSS ที่กำหนดไว้ที่ไฟล์ app.component.css

**จัดรูปแบบด้วย CSS**

ในการจัดรูปแบบ สิ่งแรกคือ การจัดเลย์เอาท์ (Layout) ก่อน ซึ่ง ใช้ลักษณะจัดด้วย CSS ได้หลายแบบ เช่น ใช้ตาราง ใช้กริด (graid) สำหรับในที่นี้เลือกใช้วิธีกริด เพราะมีความยืดหยุ่นกว่า

เพื่อให้ได้โครงร่างของเว็บ ตามที่ได้ออกแบบไว้ก่อนหน้านี้ เมื่อมองในลักษณะกริดจะเห็นว่า มีลักษณะ สามแถว คือ แถวแรกเก็บ <header> แถวที่สองเก็บ ข่าว และรูปภาพ และแถวสุดท้ายเก็บ <footer> โดยมีคลาส grid-container เป็นตัวควบคุม สำหรับแถวกลางที่เก็บรายการข่าว และรูปภาพ ต้องการให้เป็นสองคอลัมน์ ใช้คลาส grid-content เป็นตัวควบคุม ดังนั้นตามโครงสร้างนี้ จะมีรูปแบบ  ดังนี้

**Code 7.** src/app/app.component.html

```
<div class="grid-container">
      <header></header>
      <div class="grid-content">
            <div>News</div>
            <div>Picture</div>
      </div>
      <footer></footer>
</di>
```

เพื่อการควบคุมตามแนวคิดดังกล่าว จึงเขียนส่วนควบคุมตามแบบกริด เก็บไว้ใน app.component.css เพื่อให้ครอบคลุมสำหรับ app.component.html เริ่มจาก การสร้างคลาส grid-container

**Code 8.** src/app/app.component.css

```
.grid-container{
      display:grid;
      grid-template-rows: auto 1fr auto;
      grid-template-columns: 100%;
}
```

ในตัวอย่าง CSS นี้ ใช้การแสดงผล(display) แบบกริดในรูปแบบแถว(grid-template-rows) มีแถวแรกและแถวสุดท้ายมีขนาดอัตโนมัติ (auto) ตามขนาดข้อมูลที่ใช้ ส่วนแถวที่สอง มีขนาดพื้นเติมพื้นที่เหลือทั้งหมด (1fr) และคอลัมน์ใช้เต็มพื้นที่ (100%) แสดงว่ามีคอลัมน์เดียว

สำหรับแถวที่สอง ยังแยกได้เป็นกริดย่อย คือ มีหนึ่งแถว และแบบเป็นสองคอลัมน์ คอลัมน์แรกเก็บ รายการข่าว และคอลัมน์ที่สองเก็บรูป โดยมีคลาส grid-content เป็นตัวควบคุม จึงเขียนส่วนควบคุมคลาสได้ว่า

**Code 9.** src/app/app.component.css

```
.grid-content{
	display:grid;
	grid-template-columns: 50% 50%;
}
```

ในตัวอย่าง CSS นี้ ใช้การแสดงผลแบบกริด (display) แล้วให้ รูปแบบคอล์ลัมน์(grid-template-columns) สองคอลัมน์ ขนาดอย่างละครึ่งเท่าๆ กัน (50%)

เฉพาะส่วนของ คลาส grid-item ซึ่งเก็บรายการข่าว และรูป ต้องการให้มีระยะห่างระหว่างกัน (margin) ที่ 10px และระยะห่างภายใน (padding) ที่ 10px นอกจากนี้ต้องการกำหนดส่วน พื้นหลัง (background-color) เป็นสีโทนเทา #eee และมุมของขอบเป็นวงโค้ง (border-radius) ที่ 10px จึงเขียนส่วนควบคุมคลาสได้ว่า

**Code 10.** src/app/app.component.css

```
.grid-item{
	margin: 10px;
	padding:10px;
	background-color:#eee;
	border-radius:10px;

}
.grid-item img{
	width:100%;
}
```

ในตัวอย่าง CSS นี้ ยังกำหนดรูปแบบ ให้มีขนาดเต็มพื้นที่ ได้ด้วยการกำหนดความกว้าง (width) เป็น 100%

สำหรับรายละเอียดอื่น เช่น การหนดให้แสดงข้อความให้อยู่กึ่งกลางของหน้า ได้ด้วยการกำหนด text-align ให้เป็น center ซึ่งใช้กับ <footer> และ <header> รูปแบบ CSS ทั้งหมด จะเขียนในไฟล์ app.component.css ดังตัวอย่างต่อไปนี้

**Code 11.** src/app/app.component.css

```
html, body{
	width: 100%;
	height: 100%;
}
header{
	text-align:center
}
.grid-container{
	display:grid;
	grid-template-rows: auto 1fr auto;
	grid-template-columns: 100%;
}
.grid-content{
	display:grid;
	grid-template-columns: 50% 50%;
}
```

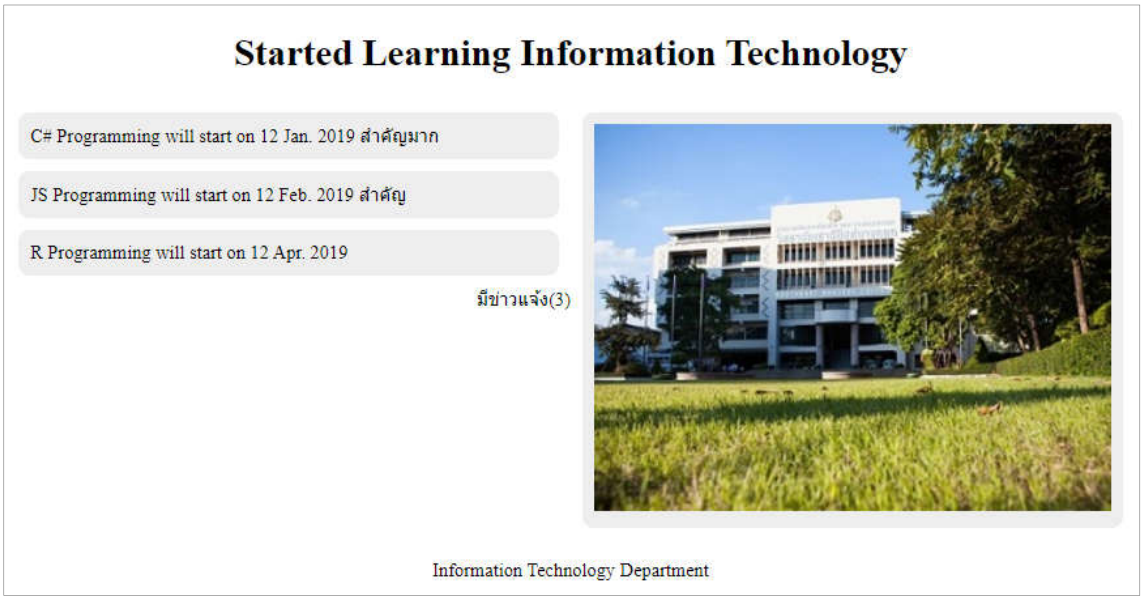
```
.grid-item{
	margin: 10px;
	padding:10px;
	background-color:#eee;
	border-radius:10px;

}
.grid-item img{
	width:100%;
}
footer{
	text-align:center;
}
```

ดังนั้นแล้วเมื่อนำคลาส CSS มาใช้กับหน้าเว็บ จะการใช้ตามตัวอย่างต่อไปนี้

**Code 12.** src/app/app.component.html

```
<div class="grid-container">
 <header>
   <h1>Started Learning Information Technology</h1>
 </header>
 <div class="grid-content">
	<div>
		<div  class="grid-item" *ngFor="let item of news">
		  {{item.content}}
		  <span [ngSwitch]='item.important'>
			<span *ngSwitchCase=1>สำคัญมาก</span>
			<span *ngSwitchCase=2>สำคัญ</span>
			<span *ngSwitchDefault></span>
		  </span>
		</div>
		<div style='text-align:right' *ngIf='news.length>0 else emptyNews'>
			มีข่าวแจ้ง({{news.length}})
		</div >
		<ng-template #emptyNews>
			ขณะนี้ไม่มีข่าว
		</ng-template>
	</div>
	<div class="grid-item" >
		<img src="./assets/images/sbc_building.jpg">
	</div>
 </div>
 <footer>
	<p>Information Technology Department</p>
 </footer>
</div>
```

การใช่ CSS นอกจากใส่เป็นไฟล์แยกต่างหาก ดังแสดงให้เห็นแล้ว ยังสามารถใส่ได้ในไฟล์ HTML โดยตรง จากตัวอย่างนี้ จะเห็นว่ามีการใช้ CSS กับ ส่วนแสดงจำนวนข่าว ให้แสดงข้อมูลชิดขวา ดังเป็น CSS ไว้ text-align




**รูป 4** หน้าเว็บที่กำหนด CSS แล้ว

**สร้างฟังก์ชันในคอมโพเน้นท์**

คลาส AppComponent ณ ตอนนี้ยังไม่มีฟังก์ชันอะไร นอกจากมีคอนสตรักเตอร์ ตอนนี้เราต้องการจะสร้างฟังก์ชันเพิ่ม เพื่อให้กรองข้อมูลที่ข่าว ที่มี active เป็น true เท่านั้น ในการนี้ ให้เพิ่มฟังก์ชัน ชื่อ getAcitveNews( ) ซึ่งจะคืนค่า อาร์เรย์ของออบเจ็กต์ News ภายในฟังก์ชัน ให้ใช้การวนซ้ำ for .. of เพื่อกรองข้อมูลที่ต้องการ ให้เขียนดังตัวอย่างต่อไปนี้เพิ่มลงในไฟล์ app.component.ts

**Code 13.** src/app/app.component.ts

```
public getActiveNews():News[]{
        var news_active = new Array();
        for(let item of this.news){
              if(item.active) news_active.push(item);
        }
        return news_active;
}
```

เพื่อให้มีการแสดงผลได้เฉพาะข้อมูลข่าวที่มี active เป็น true ดังนั้นจะให้แก้ข่าวบางตัวให้เป็น ให้ active เป็น false และจะต้องเปลี่ยนข้อมูลจากเดิมที่ใช้อ่านจาก news ให้เปลี่ยนมาอ่านจากข้อมูล getActiveNews( ) แทน โดยแก้ไขไฟล์ app.component.html ดังตัวอย่างต่อไปนี้

**Code 14.** src/app/app.component.html

```
<div class="grid-item" *ngFor="let item of getActiveNews()">
      {{item.content}}
</div>
```

เพื่อเป็นการทดสอบผลการทำงานให้แก้ไขรายการข่าว หรือออบเจ็กต์ข่าวในอาร์เรย์ให้มีตัวหนึ่งเป็น active เป็น false ให้ข่าวสุดท้ายเป็น false และดูผล หากไม่มีอะไรผิดพลาด รายข่าวที่แก้เป็น false จะไม่มีการแสดงผล

**สร้าง CSS จากคำสั่งไดเร็กทีฟ**

คำสั่งไดเร็กทีฟ เราเคยได้พบ มาบ้างแล้ว เช่น ngIf, ngFor, ngSwitch สั่งเหล่านี้จัดเป็นประเภท ไดเร็กทีฟเชิงโครงสร้าง (Structural Directives) ใน Angular มีคำสั่งไดเร็กทีฟอีกประเภทหนึ่ง เรียกว่า แอททริบิวท์ไดเร็กทีฟ (Attribute Directive) หรือไดเร็กทีฟเชิงคุณสมบัติ เพราะใช้กำหนดคุณสมบัติให้กับอิลีเม้นท์ต่าง ๆ เช่น กำหนดคลาส (NgClass) กำหนดสไตล์ (NgStyle) และ กำหนดการผูกข้อมูลแบบสองทาง (NgModel) นอกจากนี้เรายังสามารถสร้าง แอททริบิวท์เชิงคุณสมบัติได้เอง

จากตัวอย่างหน้าเว็บที่ผ่านมา ได้กำหนด CSS ให้จำนวนข่าว โดยกำหนด CSS ภายในบรรทัด HTML โดยตรง แต่เรากำหนดสไตล์ผ่านการกำหนดค่าคุณสมบัติ [style] ได้เช่นกัน เช่น

```
<div [style.text-align]="'right'">
```

การกำหนดค่าแบบนี้กำหนดได้โดยไม่ใช้ NgStyle โดยตรง รวมทั้งใส่โลจิกร่วมด้วยก็ได้ เช่น ถ้าข่าวใดสำคัญมาก ซึ่งมี important เป็น 1 ให้มีค่าสีอักษรเป็นแดง

```
<span [ngSwitch]='item.important'[style.color]="item.important==1?'red':''">
```

ถ้าหากว่ามีกำหนดสไตล์หลายคุณสมบัติ [ngStyle] แทน แต่ก็ยังใส่โลจิกได้ จากตัวอย่างต่อไปนี้ ใช้ กำหนดโลจิกข่าวที่สำคัญ เท่ากับ 1 , 2, 3 มีขนาดอักษรใหญ่มาก (24px) สำคัญบานกลาง (18px) และเล็ก(12px) โดยใช้ โลจิก if อย่างสั้น นอกจากนี้ยังกำหนด คุณสมบัติ สีอักษร และสีพื้นหลัง

**Code 15.** src/app/app.component.ts

```
importStyle:{ };
getImportStyle(important:number){
	return this.importStyle = {
		'font-size': important==1 ? '24px' : important==2?'18px':'12px',
		'color':important==1?'#FFFFFF':'',
		'background-color':important==1?'#000000':''
	};
}
```

จากฟังก์ชัน getImportStyle( ) มีตัวแปรเข้าเป็นเลขความสำคัญของข่าว ซึ่งจะคือค่า สไตล์ในชื่อ importStyle เมื่อได้ฟังก์ชันแล้ว ต่อไปต้องใส่ [ngStyle] ในส่วน รายการข่าวที่แสดงผล ดังตัวอย่างต่อไปนี้

**Code 16.** src/app/app.component.html

```
<div  class="grid-item"
	*ngFor="let item of getActiveNews()"
	[ngStyle]="getImportStyle(item.important)">
		{{item.content}}
	<span [ngSwitch]='item.important'[style.color]="item.important==1?'red':''">
		<span *ngSwitchCase=1 >สำคัญมาก</span>
		<span *ngSwitchCase=2>สำคัญ</span>
		<span *ngSwitchDefault></span>
	</span>
</div>
```


**รูป 5** หน้าเว็บที่กำหนด [ngStyle] แล้ว

จากรูปที่แสดงนี้ ข่าวที่ 3 กำหนดค่าคุณสมบัติ active ให้เป็น true เพื่อให้ได้ข่าวทั้งสามแสดงผลร่วมด้วย ซึ่งจะเห็นว่า ทั้งสามข่าว มีรูปแบบแสดงผลที่ต่างกัน โดยเฉพาะขนาดอักษร ที่มีสามระดับ

สำหรับ [ngClass] สามารถใช้แทน class ใน HTML ได้เลย แต่มีต่างเล็กน้อยคือต้องเพิ่มเครื่องหมาย ' ' คล่อมชื่อคลาสก่อน เช่น

```
<div [ngClass]="'grid-content'">
```

นอกจากจะเพิ่มคลาส แล้วยังประกาศในลักษณะลดหรือไม่มีคลาสได้ ด้วยการใส่โลจิก if ได้ เช่น ถ้าจริง ก็ให้มีชื่อคลาสที่กำหนดได้ แต่ถ้าไม่จริง ก็ไม่ต้องมีคลาส

```
<div [ngClass]="true ? 'grid-content':''">
```

**สร้างไดเร็กทีฟขึ้นใช้เอง**

ที่ผ่านมา เราใช้ ไดเร็กทีฟเชิงคุณสมบัติที่มีมาให้แล้วใน Angular ถ้าหากว่ายังรู้สึกไม่เพียงพอ เราก็สามารถสร้างขึ้นใช้งานได้เอง เช่น เราต้องการจะสร้าง สไตล์ ให้อักษรชิดขวา โดยสร้างไดเร็กทีฟ ชื่อ appTextRight จะมาจากการสร้างไดเร็กทีฟดังนี้

```
ng generate directive textRight
```

เมื่อสร้างไดเร็กทีฟนี้แล้ว จะมีการเพิ่ม ดีเร็กทีฟนี้ในโมดูล (app.module.ts) อัตโนมัติ และสร้างไฟล์ text-right.directive.ts มาให้มาด้วย

ในกรณีนี้เราต้องการ กำหนดอักษรชิดขวา ต้องใช้ ElementRef เพื่ออ้างอิงอีลีเม้นท์ของ HTML และใช้กำหนดสไตล์ จึงต้องนำเข้า และปรับปรุงไฟล์ใหม่ดังนี้

**Code 17.** src/app/text-right.directive.ts

```
import { Directive, ElementRef } from '@angular/core';

@Directive({
```

```
        selector: '[appTextRight]'
})
export class TextRightDirective {
       constructor(el: ElementRef) {
            el.nativeElement.style.textAlign='right';
       }
}
```

เมื่อสร้างไดเร็กทีฟ TextRightDirective แล้ว การนำมาใช้ต้องนำเข้าด้วย เช่น ต้องการใช้กับ Appcomponent

```
import { TextRightDirective } from './text-right.directive';
```

การนำไปใช้ กับ HTML เพียงเพิ่มคุณสมบัตินี้กับอิลีเม้นท์ที่ต้องการให้มีอักษรชิดขวา เช่น ต้องการใช้ ส่วนนำจำนวนข่าว จากเดิมใช้แนว `[style.text-align]` แต่ตอนนี้ใช้ ไดเร็กทีฟแทน

```
<div appTextRight>
```

คุณสมบัติที่มียังเพิ่มได้อีก เช่น ต้องการเพิ่มการรองรับเหตุการณ์ การตอบสนองกับเม้าท์ เช่น ถ้าต้องการให้ เมื่อเม้าท์เลื่อนเข้าอิลีเม้นท์ที่ต้องการ ให้เปลี่ยน การจัดตำแหน่งไปอยู่ทางซ้าย และเมื่อเลื่อนเม้าท์ออก ให้เคลื่อนกลับมาอยู่ที่เดิม

การทำให้รองรับเหตุการณ์ได้ จะต้องนำเข้า HostListener เพิ่มอีกตัวหนึ่ง และเพิ่มฟังก์ชัน เหตุการณ์ onMouseEnter( ), onMouseLeave( ) และ moveLeft( ) โดยมีสองฟังก์ชันแรกนั้นเรียกใช้ฟังก์ชันที่สามอีกที

**Code 18.** src/app/text-right.directive.ts

```
import { Directive, ElementRef, HostListener } from '@angular/core';

@Directive({
  selector: '[appTextRight]'
})
export class TextRightDirective {
  constructor(private el: ElementRef) {
      this.el.nativeElement.style.textAlign='right';
  }
  @HostListener('mouseenter')
  onMouseEnter() {
        this.moveLeft('left');
  }
  @HostListener('mouseleave')
  onMouseLeave() {
      this.moveLeft('right');
  }
  private moveLeft(move:string){
      this.el.nativeElement.style.textAlign=move;
  }
}
```

จากตัวอย่างนี้ ให้สังเกตว่า ฟังก์ชัน moveLeft( ) อ้างอิง this.el ซึ่งถือเป็นสมาชิกตัวหนึ่งของคลาสนี้ จึงต้องดัดแปลงการประกาศค่าใน constructor( ) ด้วย เพื่อให้ el เป็นสมาชิกตัวหนึ่งของคลาสนี้

**ไปป์ (Pipe)**

นอกจากการแสดงผลในรูปแบบสไตล์ที่ทำศึกษามาก่อนหน้านี้ ยังมีรูปแบบการแสดงผลที่ส่งต่อกันได้ ที่เรียกว่า ไปท์ การส่งต่อเพื่อการแสดงผลในรูปแบบที่เปลี่ยนไปจากเดิม เช่น วันที่ ที่มักอยู่ในรูปภาษาอังกฤษ แต่ต้องให้อยู่ในรูปแบบภาษาไทย หรือการจัดเรียงลำดับ วัน เดือน ปี

ยกตัวอย่างเช่น ในคลาส AppComponent ให้เพิ่มสมาชิก ชื่อ create มีวันที่ ตามตัวอย่างต่อไปนี้ ซึ่งตัวแปร create นี้ต่อไป สามารถนำไปแสดงในหน้าเว็บได้

```
create = new Date(2020, 0, 13);  //Jan 13, 2020
```

การนำไปแสดงใช้การ ไปท์ แสดงในรูปแบบวันที่ ใน HTML เช่น

```
{{create | date }}
```

จากตัวยอย่างนี้ จะใช้เครื่องหมาย | แทนการไปป์ การแสดงผล จะแสดง ในรูปแบบ เดือน วัน ปี เราสามารถกำหนดรูปแบบที่ต่างจากนี้ได้

| | | |
|---|---|---|
| `{{create \| date : "MM dd yy" }}` | แสดง | 01/13/20 |
| `{{create \| date : "MMMM dd, yyyy" }}` | แสดง | January 13, 2020 |
| `{{create \| date : "short" }}` | แสดง | 1/13/20, 12:00 AM |
| `{{create \| date : "fullDate" \| uppercase }}` | แสดง | `MONDAY, JANUARY 13, 2020` |

รูปแบบเหล่านี้ใช้ได้เหมือนกับภาษา JavaScript แต่อย่างไรก็ตาม รูปแบบเหล่านี้ไม่เป็นภาษาไทย การสร้างรูปไปท์ขึ้นเอง เช่น ต้องการ ไปท์ ที่มีรูปแบบชื่อ thaiDate เริ่มจาก สร้างคลาส ThaiDatePipe ดังนี้

```
ng generate pipe ThaiDate
```

เมื่อได้คลาสใหม่มาแล้ว สิ่งที่ต้องเพิ่ม คือ PipeTransform เมื่อใช้ฟังก์ชัน transform( ) ภายใต้ฟังก์ชันนี้ รับตัวแปรเป็น Date ซึ่งต่อไป นำไปดำเนินการ แปลงรูปแบบให้เป็นภาษาไทย จัดเรียงลำดับตาม วัน เดือน ปี

**Code 19.** src/app/thai-date.pipe.ts

```
import { Pipe, PipeTransform } from '@angular/core';

@Pipe({
  name: 'thaiDate'
})
export class ThaiDatePipe implements PipeTransform {

  transform(value: Date): string {
       var date : number = value.getDate();
       var month : string;
       var year :number = (value.getFullYear() + 543) ;
       switch (value.getMonth()){
              case 1: month="มกราคม";break;
              case 2: month="กุมภาพันธ์";break;
```

```
            case 3: month="มีนาคม";break;
            case 4: month="เมษายน";break;
            case 5: month="พฤษภาคม";break;
            case 6: month="มิถุนายน";break;
            case 7: month="กรกฎาคม";break;
            case 8: month="สิงคาคม";break;
            case 9: month="กันยายน";break;
            case 10: month="ตุลาคม";break;
            case 11: month="พฤษจิกายน";break;
            default: month="ธันวาคม";break;
        }
      return date + " "+ month +" "+year;
    }
}
```

จากรูปแบบวันที่เป็นภาษาไทยที่ได้สร้าง การนำไปใช้ต้อง นำเข้าคลาสนี้ก่อน นำเข้าในไฟล์ app.component.ts

```
import { ThaiDatePipe } from './thai-date.pipe';
```

การนำไปใช้ก็ให้ ไปป์ ในชื่อที่สร้างไว้ เช่น

```
{{create | thaiDate }}
```
แสดงผล 13 ธันวาคม 2563

**การทำงานร่วมกับ Bootstrap**

การเพิ่มไลบรารีเพิ่มเติมให้กับ Angular อย่างในกรณีที่ต้องการจะใช้ Bootstrap เป็นไลบรารีที่ใช้จัดแต่งหน้าเว็บ และใช้งาน JQuery ร่วมอยู่นั้น จำเป็นต้องติดตั้งไลบรารีเหล่านี้เข้าไปโมดูล ซึ่งทำได้โดย การอ้างอิงไลบรารี เว็บบริการไลบรารี หรือ อ้างอิงจากที่เก็บไว้ที่เว็บแอปฯของตนเอง

สำหรับวิธีแรกถือว่าทำได้ง่ายที่สุด เพียงใส่การอ้างอิงที่หน้าเว็บเริ่มต้น ซึ่งคือ index.html โดยเลือกเว็บที่มีบริการไฟล์ไลบรารี เช่น ตัวอย่างต่อไปนี้เลือกจากเว็บ maxcdn.bootstrapcdn.com

**Code 20.** src/index.html

```
<!doctype html>
<html lang="en">
<head>
  <meta charset="utf-8">
  <title>MyAngular</title>
  <base href="/">
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <link rel="icon" type="image/x-icon" href="favicon.ico">
  <!-- Latest compiled and minified CSS -->
  <link rel="stylesheet"
   href="https://maxcdn.bootstrapcdn.com/bootstrap/4.4.1/css/bootstrap.min.css">
</head>
<body>
```

```
  <app-root></app-root>
  <script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.4.1/jquery.min.js"></script>
 <script
src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.16.0/umd/popper.min.js">
 </script>
  <script
src="https://maxcdn.bootstrapcdn.com/bootstrap/4.4.1/js/bootstrap.min.js">
  </script>
</body>
</html>
```

ส่วนอีกวิธีหนึ่ง คือการเก็บไฟล์ไว้ที่เว็บตนเอง วิธีนี้ก็มีข้อดีคือ ไม่ต้องต่อเชื่อมกับอินเทอร์เน็ตก็สามารถทำงานได้ แต่ขั้นก็ทำหลายขั้นตอน เริ่มจาก การติดตั้งไลบรารีก่อน (คำสั่งต่อไปนี้ต้องใช้ขณะอยู่บนไดเร็กทอรีของโปรเจ็กต์ (myAngular) ที่กำลังทำงานอยู่)

**Code 21.**

```
npm install jquery --save
npm install popper.js --save
npm install bootstrap --save
```

หลังจากนั้นแก้ไขไฟล์ angular.json ซึ่งไฟล์นี้อยู่นอกสุดของโฟลเดอร์งาน ไฟล์ที่ระบุนี้หากเปิดดูตามเส้นทางจะมีไฟล์เหล่านี้อยู่จริง ใน node_modules ให้เพิ่ม jquery ในส่วน scripts ของ projects> myAngular> architect> build> builder> options > scripts

**Code 22.** myAngular/angular.json

```
"scripts": [
"./node_modules/jquery/dist/jquery.slim.js",
"./node_modules/popper.js/dist/umd/popper.js",
"./node_modules/bootstrap/dist/js/bootstrap.js" ]
```

และสไตล์ชีท

**Code 23.** myAngular/angular.json

```
"styles":[
"src/styles.css",
"./node_modules/bootstrap/dist/css/bootstrap.css",
],
```

หลังนั้นให้เปิดการทำงานใหม่ (ng serve --open) อีกครั้ง หากเปิดมาก่อนก็ให้ปิด แล้วเปิดทำงานใหม่ เพราะต้องเริ่มต้นแปลผลใหม่ตั้งแต่ต้น ต่อไปก็นำประกาศใช้ bootstrap ใน หน้าหลัก ซึ่งคือ index.html ซึ่งไม่ต้องใส่ ไฟล์ของ Bootstrap และ JQuery อีกต่อไป

**Code 24.** src/index.html

```
<!doctype html>
<html lang="en">
<head>
  <meta charset="utf-8">
  <title>MyAngular</title>
```

```
  <base href="/">
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <link rel="icon" type="image/x-icon" href="favicon.ico">
</head>
<body>
  <app-root></app-root>
</body>
</html>
```

**สร้างกริดตามรูปแบบ Bootstrap**

เมื่อเชื่อมไฟล์ต่างของ Bootstrap แล้ว ต่อมาก็มาลองทดสอบ ใช้คุณสมบัติ CSS ของ Bootstrap กัน ในที่นี้ได้ยกตัวอย่างการสร้างกริด ในรูปแบบ แถว และคอลลัมน์ เลี่ยนแบบหน้าเดิมที่สร้างด้วยกริดก่อนหน้านี้

**Code 25.** src/app/app.component.html

```
<div class="grid-container">
 <header>
   <h1>Started Computer Learning at SBC </h1>
 </header>
 <div class="container">
    <div class='row'>
             <div class='col'>
                    <div class='row'*ngFor="let item of getActiveNews()">
                    {{item.content}}
                    </div>
             </div>
             <div class="col">
                    <img src="./assets/images/sbc_building.jpg">
             </div>
      </div>
 </div>
 <footer>
      <p>Southeast Bangkok College: Information Technology Department
      ({{create | thaiDate}})</p>
 </footer>
</div>
```


**รูป 6** หน้าเว็บที่กำหนด CSS ด้วย Bootstrap

**การทำงานร่วมกับ JQuery**

เหลืออีกสิ่งหนึ่ง คือการทำงานร่วมกับ JQuery ถึงแม้ Angular จะมีสคริบท์ภาษาของตนเอง แต่บางทีก็ยังไม่พอ อยากใช้ JQuery ช่วยเสริมอีกตัวหนึ่ง

ตัวอย่างต่อไปนี้ เป็นตัวอย่างที่ใช้กับ คอมโพเน้นท์หนึ่ง เป็นการเรียกใช้งาน JQuery แต่ก่อนใช้งานจะต้องนำเข้าไลบรารีก่อน

**Code 26.** src/app/app.component.ts

```
import * as $ from 'jquery';
```

ส่วนการเรียกใช้ ใช้ภาษาเหมือนภาษา JavaScript ได้เลย ตัวอย่างเช่น การใช้ เลือก id=toggle-image ผูกกับเหตุการณ์คลิก กับฟังก์ชัน โดยสมมติว่า กำหนดให้มีตัวชี้เป็นรูปมือ เพื่อให้รู้ว่าคลิกได้ ต่อมากำหนดฟังก์ชันคลิก เมื่อคลิกแต่ละครั้งจะสลับเปลี่ยนรูปไป-มา ซึ่งมีอยู่สองรูป สองรูปนี้เก็บตามเส้นทางที่กำหนดนี้ ดังตัวอย่างต่อไปนี้ ดังมีสองไฟล์ที่ต้องแก้ไข

**Code 27.** src/app/app.component.ts

```
ngOnInit(): void {
    $('#toggle-image').css('cursor', 'pointer');
    $('#toggle-image').click(function(){
        let img1:string='./assets/images/sbc_building.jpg';
        let img2:string='./assets/images/sbc_building2.jpg';
        let src:string = ($('#toggle-image').attr('src') === img2)
            ? img1
            : img2;
         $('#toggle-image').attr('src', src);
    });
}
```

**Code 28.** src/app/app.component.html

```
<img id='toggle-image' src="./assets/images/sbc_building.jpg">
```

**สรุป**

ถึงตอนนี้ การทำงานกับคอมโพเน้นท์เบื้องต้น ที่ใช้การอ่านข้อมูล จากอาร์เรย์ ทั้งที่สร้างขึ้นเองจากออบเจ็กต์ JSON และคลาส ที่กำหนดค่าเริ่มต้นในคอนสตรักเตอร์ของคอมโพเน้นท์หลัก ข้อมูลที่กำหนดขึ้นมานี้ นำไปแสดงผลที่ไฟล์ HTML ด้วยการผูกข้อมูลด้วยเครื่องหมายปีกคู่ ( {{ ... }} ) ซึ่งเป็นผูกข้อมูลแบบบทางเดียว การแสดงข้อมูลของหน้าเว็บ มีโลจิก 3 รูปแบบ คือ การวนซ้ำใช้คำสั่ง *ngFor การเลือกใช้ *ngIf และ [ngSwitch] สำหรับการวนซ้ำในไฟล์ ts ได้มีการสร้างฟังก์ชันภายในคลาส AppComponent โดยใช้คำสั่ง for... of เพื่อวนอ่านค่าในอาร์เรย์ นอกจากนี้ยังได้เรียนรู้การใช้ CSS ด้วยการใช้งานพื้นฐาน โครงการจัดเรียงหน้าเว็บด้วยกริด การจัดรูปแบบด้วยคำสั่งไดเร็กทีฟ การไปป์ และการใช้งานร่วมกับ Bootstrap และ JQuery ขั้นพื้นฐาน

ยังมีการผูกข้อมูลแบบสองทาง ซึ่งเป็นจุดเด่นของ Angular ที่ช่วยให้การทำงานกับฟอร์ม (Form) ของ HTML ทำงานได้ง่ายขึ้น ดังจะศึกษาต่อไป เรื่องฟอร์ม และอีเว้นท์ (Event) ของฟอร์ม

**อ้างอิง**

Techiediaries. (March 25, 2020). 3+ Ways to Add Bootstrap 4 to Angular 9/8 With Example & Tutorial. https://www.techiediaries.com/angular-bootstrap/.