

พัฒนา Web App แบบ Responsive

ด้วย Bootstrap5

ร่วมกับ Vue.js และ Node.js



ปรับแต่งเว็บเพจพร้อมการแสดงผล
แบบ Responsive



ควบคุมการแสดงผลเนื้อหาแบบไดนามิก
ตามเงื่อนไขที่ต้องการ



รับส่งและประมวลผลข้อมูลทางฝั่ง
เซิร์ฟเวอร์ผ่าน REST API

SE-ED
inspiration starts here



ดาวน์โหลดโค้ดหนังสือเล่มนี้ได้ที่
<http://www.developerthai.com>

บัญชา ปะสีละเตสัง

ผู้เขียนหนังสือขายดีระดับ Best Seller
ด้าน Programming

พัฒนา Web App 1100 Responsive ด้วย Bootstrap5 ร่วมกับ Vue.js และ Node.js

โดย บัญชา ปะสีละเตลัง

สงวนลิขสิทธิ์ตามกฎหมาย โดย บัญชา ปะสีละเตลัง © พ.ศ. 2565

ห้ามคัดลอก ลอกเลียน ดัดแปลง ทำซ้ำ จัดพิมพ์ หรือกระทำการอื่นใด โดยวิธีการใดๆ ในรูปแบบใดๆ

ไม่ว่าส่วนหนึ่งส่วนใดของหนังสือเล่มนี้ เพื่อเผยแพร่ในสื่อทุกประเภท หรือเพื่อวัตถุประสงค์ใดๆ

นอกจากจะได้รับอนุญาต

ข้อมูลทางบรรณานุกรมของหอสมุดแห่งชาติ

บัญชา ปะสีละเตลัง.

พัฒนา Web App แบบ Responsive ด้วย Bootstrap5 ร่วมกับ Vue.js และ Node.js.---กรุงเทพฯ : ซีเ็ดยูเคชั่น, 2565.

436 หน้า.

1. โปรแกรมประยุกต์บนเว็บ. 2. โปรแกรมประยุกต์บนอุปกรณ์เคลื่อนที่.

I. ชื่อเรื่อง.

005.355

Barcode (e-book) : 9786160843831

ผลิตและจัดจำหน่ายโดย



บริษัท ซีเ็ดยูเคชั่น จำกัด (มหาชน)
SE-EDUCATION PUBLIC COMPANY LIMITED

เลขที่ 1858/87-90 ถนนเพชรตัด แขวงบางนาใต้ เขตบางนา กรุงเทพฯ 10260

โทรศัพท์ 0-2826-8000

[หากมีคำแนะนำหรือติชม สามารถติดต่อได้ที่ comment@se-ed.com]



คำนำ

การสร้างเว็บแอปพลิเคชันให้ครบถ้วนสมบูรณ์แบบ จนสามารถใช้งานได้จริงนั้น อาจมีหลายขั้นตอนที่เราดำเนินการร่วมกัน ซึ่งในหนังสือเล่มนี้ ได้รวบรวมสิ่งที่จำเป็นต้องรู้ทั้งหมดสำหรับการพัฒนาเว็บแอปพลิเคชันเอาไว้อย่างครบถ้วน นับตั้งแต่การจัดโครงสร้างของเพจด้วย HTML5 การปรับแต่งรูปแบบการแสดงผลให้สวยงามน่าใช้งาน พร้อมทั้งรองรับการแสดงผลบนหน้าจอที่มีขนาดแตกต่างกันด้วย CSS3 และ Bootstrap5 รวมถึงการควบคุมเนื้อหาให้ตรงตามเงื่อนไขด้วย Vue.js ตลอดจนการรับส่งข้อมูลและประมวลผลทางฝั่งเซิร์ฟเวอร์โดยใช้ Node.js พร้อมทั้งไลบรารีที่เกี่ยวข้องอีกหลายอย่าง

ดังนั้น จึงหวังว่าผู้อ่านจะสามารถนำความรู้ทั้งหมดที่ได้จากหนังสือเล่มนี้ทั้งส่วนของ Bootstrap5, Vue.js และ Node.js ไปประยุกต์ใช้ร่วมกันในการพัฒนาเว็บแอปพลิเคชันที่สามารถทำงานได้อย่างถูกต้องในทุกขั้นตอนและใช้งานได้จริงตามที่ต้องการ

บัญชา ปะสีละเตสัง

banchar.pa@yahoo.com

facebook.com/developerthai

<http://www.developerthai.com>



```
<div class="d">
  <div class="me-4">
    <h6>flex-column</h6>
    <div class="d-inline-flex">
      <div class="p-2 mb-2 bg-light">
        <div class="p-2 mb-2 bg-light">
          <div class="p-2 mb-2 bg-light">Item 1</div>
        </div>
      </div>
    </div>
    <div>
      <h6>flex-column-reverse</h6>
      <div class="d-inline-flex flex-column">
        <div class="p-2 mb-2 bg-light">Item 1</div>
        <div class="p-2 mb-2 bg-light">Item 2</div>
        <div class="p-2 mb-2 bg-light">Item 3</div>
      </div>
    </div>
  </div>
```

SEEDED
inspiration starts here



บทที่ 1 VS Code และพื้นฐาน HTML5	15
การติดตั้งและใช้งาน VS Code.....	15
● การเปลี่ยนรูปแบบสีของ VS Code.....	16
● การย่อและขยายขนาด (Zoom Out & Zoom In).....	17
● การกำหนดรูปแบบฟอนต์ของ Code Editor.....	17
● การติดตั้งส่วนขยายของ VS Code.....	18
การทดสอบเว็บเพจด้วย Live Server	19
ลักษณะของแท็กและอิลิเมนต์	21
แอตทริบิวต์ของแท็ก	22
โครงสร้างหลักของ HTML.....	22
กลุ่มแท็กที่ส่วนหัวของเว็บเพจ.....	23
การกำหนดภาพ Favicon.....	25
กลุ่มแท็กสำหรับจัดแบ่งเนื้อหา	26
กลุ่มแท็กสำหรับจัดรูปแบบข้อความ.....	28
แท็กสำหรับแสดงคำอธิบาย	29
อักขระพิเศษของ HTML	29
แท็กสำหรับแสดงรายการ.....	30
การเชื่อมโยงด้วยลิงก์	33

การแสดงผลภาพบนเว็บเพจ	35
การเล่นไฟล์มัลติมีเดียบนเว็บเพจ	36
การสร้างตาราง	37

บทที่ 2 พื้นฐานที่สำคัญของ CSS..... 39

พร็อพเพอร์ตี้ของ CSS	39
• การกำหนดค่าของพร็อพเพอร์ตี้	39
• พร็อพเพอร์ตี้พื้นฐานที่ควรรู้จัก	40
ขอบเขตของการกำหนดสไตล์แบบต่างๆ	41
• การกำหนดสไตล์แบบ Inline.....	41
• การกำหนดสไตล์แบบ Internal.....	41
• การกำหนดสไตล์แบบ External.....	43
ซีเลกเตอร์รูปแบบต่างๆ	44
การเขียนคำอธิบายใน CSS	51
ลำดับความสำคัญของสไตล์	51
การกำหนดความสำคัญด้วย !important.....	52
การกำหนดสี (Color)	53
การกำหนดหน่วยแบบต่างๆ (Unit)	54
การกำหนดสไตล์เกี่ยวกับฟอนต์ (Font)	55
การกำหนดสไตล์เกี่ยวกับข้อความ (Text).....	57
การกำหนดสไตล์ของเส้นขอบ (Border).....	58
การกำหนดระยะ Padding.....	60
การกำหนดระยะ Margin	61
การกำหนดสไตล์การแสดงผล (Display).....	62

บทที่ 3 BS5: การติดตั้งและคลาสพื้นฐาน 63

การนำ Bootstrap5 มาใช้ในเว็บเพจ.....	63
• การเชื่อมโยงกับไฟล์ Bootstrap5 ผ่าน CDN.....	64
• การเชื่อมโยงกับไฟล์ Bootstrap5 โดยตรง.....	64
• แนวทางการใช้คลาสของ Bootstrap5.....	66
Contextual Color	66
รูปแบบพื้นหลัง.....	67

รูปแบบของข้อความและตัวอักษร	69
รูปแบบเส้นขอบและมุม	71
รูปแบบของตาราง.....	73
ระยะ Margin และ Padding.....	75
การแสดงผลอีลิเมนต์ (Display)	78
การใช้ภาพ SVG Icon จาก Bootstrap	79
● การเชื่อมโยงกับไฟล์ไอคอน.....	79
● การแสดงภาพไอคอน	80
บทที่ 4 BS5: การใช้คอมโพเนนต์ (1).....	83
คอมโพเนนต์ Button	83
คอมโพเนนต์ ButtonGroup	85
คอมโพเนนต์ List Group	86
คอมโพเนนต์ Pagination.....	89
คอมโพเนนต์ Badge	90
คอมโพเนนต์ Alert	92
คอมโพเนนต์ Toast	94
คอมโพเนนต์ Offcanvas.....	98
คอมโพเนนต์ Modal	101
บทที่ 5 BS5: การใช้คอมโพเนนต์ (2).....	105
คอมโพเนนต์ Accordion	105
คอมโพเนนต์ Dropdown.....	109
คอมโพเนนต์ Card.....	114
คอมโพเนนต์ Nav และ Tab.....	118
คอมโพเนนต์ Navbar	123
บทที่ 6 BS5: การจัดรูปแบบของฟอร์ม	129
ลักษณะของ HTML Form.....	129
การปรับแต่งอินพุตในกลุ่ม Form Control.....	131
การแสดงผลข้อความแบบ Floating Label.....	134
การจัดรูปแบบของ Select	136

การจัดรูปแบบของ Checkbox และ Radio	137
การสร้างตัวเลือกแบบ Switch.....	139
การสร้าง Button Group ด้วย Radio.....	140
การสร้าง Input Group	141

บทที่ 7 BS5: การแสดงฟล็ก Flex..... 145

Flex Container และ Flex Items	145
ทิศทางการจัดเรียง Item	146
การห่อหุ้ม Item.....	147
แกนหลักและแกนขวาง	149
การจัดเรียงเนื้อหาตามแกนหลัก.....	149
การจัดแนวขอบตามแกนขวางโดยเทียบ Flex Line.....	152
การจัดแนวขอบตามแกนขวางโดยเทียบกับ Container	155
การจัดแนวขอบตามแกนขวางแบบแยก Item	158
การกำหนดสัดส่วนการยืดออกของ Item	159

บทที่ 8 BS5: การจัดโครงสร้างแบบ Responsive..... 161

หลักการเบื้องต้นของ Grid System.....	161
การกำหนด Breakpoint.....	164
• การกำหนด Breakpoint ชุดเดียว	165
• การกำหนด Breakpoint หลายชุด	168
• Breakpoint สำหรับหน้าจอเล็กพิเศษ (Extra Small)	170
การกำหนดขนาดแบบ Auto	171
Responsive Class ที่น่าสนใจ.....	173
การจัดเรียงคอลัมน์แบบ Gutter	175
แนวทางการจัดโครงสร้างแบบ Responsive.....	177
• ตัวอย่างการแสดงรายละเอียดสินค้า	177
• ตัวอย่างการแสดงรายการเมนูบน Sidebar	179
• ตัวอย่างฟอร์มสำหรับเข้าสู่ระบบ	181
• ตัวอย่างฟอร์มที่มีอินพุตหลายชนิด.....	182

บทที่ 9	บทวน JavaScript และ Node.js	185
การติดตั้ง Node.js		185
การใช้ Terminal ของ VS Code		186
การทดสอบไฟล์ JavaScript ด้วย Node.js		188
องค์ประกอบเบื้องต้นของการเขียนโค้ด		189
• การแสดงผลด้วย console.log()		189
• การเขียนคำอธิบายโค้ด		190
• จุดสิ้นสุดคำสั่ง		190
• คำสงวนของ JavaScript		191
ตัวแปรใน JavaScript		191
ค่าคงที่		192
โอเปอเรเตอร์สำหรับการคำนวณ		193
โอเปอเรเตอร์การเปรียบเทียบและตรรกะ		193
โอเปอเรเตอร์แบบ Ternary		194
ค่า null, undefined และ NaN		194
การเปรียบเทียบค่าที่เทียบเท่า false ด้วย		195
การเชื่อมต่อสตริง		195
การแทรกค่าตัวแปรในสตริง		196
การใช้คำสั่ง if และ else		196
การใช้คำสั่ง switch-case		197
การใช้ลูปแบบ for		198
การใช้คำสั่ง break และ continue		198
การใช้ลูปแบบ while		199
การใช้ลูปแบบ do-while		199
การสร้างและใช้งานฟังก์ชัน		199
Callback Function		202
Arrow Function		203
ข้อมูลชนิดอาร์เรย์		204
เมธอดของอาร์เรย์		205
ออบเจกต์ของ JavaScript		207
ออบเจกต์ Math		208

ออบเจกต์ Number	209
ออบเจกต์ String	210
ออบเจกต์ Date.....	211
การใช้โมดูลในแบบ JavaScript.....	212
• การสร้างและส่งออกโมดูล.....	212
• การนำเข้าโมดูล.....	213
• การส่งออกแบบ default.....	214
ทบทวนการเขียนจาวาสคริปต์ในเว็บเพจ	215

บทที่ 10 Vue: การใช้งานเบื้องต้นและอีเวนต์ 219

เกี่ยวกับ Vue.js.....	219
การใช้ Vue ในแบบ SPA.....	220
• การเชื่อมโยงผ่าน CDN	220
• การเชื่อมโยงภายในเครื่องของเราเอง.....	221
การใช้งานร่วมกับ Bootstrap.....	222
การแทรกข้อมูลด้วย Mustache.....	223
การกำหนดข้อมูลด้วยเมธอด data().....	225
กำหนดวิธีดำเนินการด้วยออปชัน methods.....	227
กำหนดวิธีคำนวณด้วยออปชัน computed.....	229
การกำหนดและจัดการอีเวนต์	230
• วิธีจัดการอีเวนต์ในแบบ JavaScript	230
• วิธีจัดการอีเวนต์ในแบบ Vue.js	231
• การตรวจสอบข้อมูลจากอีเวนต์	236
• วิธีป้องกันการกระทำแบบปกติเมื่อเกิดอีเวนต์.....	237
การสร้างค่าอ้างอิงให้กับอิลิเมนต์	238

บทที่ 11 Vue: การใช้ Directive และ Binding..... 243

Directive ของ Vue.js.....	243
การกำหนดข้อมูลด้วย v-text และ v-html	244
การแสดงผลตามเงื่อนไข.....	245
• การใช้ v-show.....	245

• การใช้ v-if.....	247
• การใช้ v-else	248
• การใช้ v-else-if	250
การแสดงผลข้อมูลแบบรายการด้วย v-for	253
• การใช้ v-for กับอิลิเมนต์ที่แสดงผลซ้ำๆ	253
• การใช้ v-for ร่วมกับ template	255
การ Binding ค่าของแอตทริบิวต์	256
Class Binding	258

บทที่ 12 Vue: ใช้ V-Model ร่วมกับฟอร์ม..... 267

หลักการของ v-model.....	267
อินพุตประเภทข้อความ.....	270
อินพุตชนิด Checkbox และ Switch	275
อินพุตชนิด Radio และ Button Group	280
อินพุตชนิด Select	285
การตรวจสอบข้อมูลของฟอร์ม.....	289
• การใช้แอตทริบิวต์พื้นฐานของ HTML.....	289
• การตรวจสอบฟอร์มด้วย Vue.js.....	290

บทที่ 13 Vue: สร้างและใช้งานคอมโพเนนต์..... 297

แนวคิดเกี่ยวกับคอมโพเนนต์.....	297
การสร้างคอมโพเนนต์แบบ String Template.....	299
• การสร้างออบเจ็กต์ของคอมโพเนนต์.....	299
• หลักการตั้งชื่อคอมโพเนนต์.....	301
• ออปชันอื่นๆ ของคอมโพเนนต์.....	301
• ออปชันของ Vue Instance.....	303
การกำหนดข้อมูลของคอมโพเนนต์ด้วย props.....	303
การกำหนดอีเวนต์ให้กับคอมโพเนนต์	308
การใช้ v-model ร่วมกับคอมโพเนนต์.....	311
การใช้คอมโพเนนต์ในแบบโมดูล.....	314

บทที่ 14 **Vue: สร้างเว็บในแบบ Vue CLI** 317

การสร้างเว็บในแบบ Vue CLI.....	317
โครงสร้างของแอปแบบ Vue CLI	320
หลักการทำงานของ Vue CLI	322
Single File Component (SFC).....	324
• องค์ประกอบของคอมโพเนนต์.....	324
• การนำคอมโพเนนต์มาใช้ใน App.vue	327
• การใช้งานร่วมกับ Bootstrap	329
• การจัดเก็บและเข้าถึงไฟล์ Assets	332
การส่งข้อมูลระหว่างคอมโพเนนต์ผ่าน Event Bus	332

บทที่ 15 **Vue: กำหนดเส้นทางด้วย Router** 337

การจัดเตรียมแอปและติดตั้งโมดูล.....	337
หลักการของ Vue Router.....	338
ข้อกำหนดในแต่ละส่วนของ Vue Router.....	340
• การสร้างรายการเมนู.....	340
• การสร้าง View Component	342
• การกำหนดเส้นทางและคอมโพเนนต์ในไฟล์ index.js	344
• การกำหนดพารามิเตอร์สำหรับกรณี Error 404 (Not Found).....	346
• การปรับแต่งเมนู	347
การส่งข้อมูลไปพร้อมกับพารามิเตอร์	349
• การส่งข้อมูลแบบพารามิเตอร์.....	349
• การส่งข้อมูลแบบ Query String.....	353
การใช้ Vue Router ร่วมกับ Bootstrap.....	355

บทที่ 16 **Node: การรับส่งข้อมูลกับเซิร์ฟเวอร์**..... 359

บททวนการใช้ Node สำหรับ Web Server.....	359
บททวนการใช้ Express สำหรับ Web Server.....	361
การใช้ Vue CLI ร่วมกับ Node และ Express.....	362
• การติดตั้งไฟล์ทางฝั่ง Local (Vue CLI).....	363
• การติดตั้งไฟล์ที่ฝั่ง Server (Node/Express)	365
• การรันทดสอบฝั่ง Local และ Server.....	366

การใช้ฟังก์ชัน fetch() ทางฝั่ง Local.....	366
• รูปแบบของฟังก์ชัน fetch().....	367
• การใช้ fetch() ร่วมกับ Vue CLI.....	368
• การใช้ fetch() เมื่อเกิดอีเวนต์.....	369
• การส่งข้อมูลไปยังฝั่ง Server ในแบบ GET	370
• การส่งข้อมูลไปยังฝั่ง Server ในแบบ POST.....	371
ข้อกำหนดทางฝั่ง Server.....	371

บทที่ 17 Node: พอร์ม การอัปโหลด และ Session..... 377

การส่งข้อมูลจากฟอร์มผ่าน REST API	377
• การส่งข้อมูลจากฟอร์มด้วยเมธอด GET	377
• การส่งข้อมูลจากฟอร์มด้วยเมธอด POST	380
การอัปโหลดไฟล์.....	383
• วิธีการทางฝั่ง Local	383
• วิธีการทางฝั่ง Server.....	384
การจัดเก็บและเข้าถึงไฟล์ที่อัปโหลด.....	385
การเก็บข้อมูลแบบ Cookie.....	390
• การจัดเก็บข้อมูลแบบ Cookie.....	390
• การอ่านข้อมูลจาก Cookie.....	391
• การลบ Cookie	392
การจัดเก็บข้อมูลแบบ Session.....	395
• การจัดเก็บข้อมูลใน Session.....	396
• การอ่านค่าจาก Session.....	397
• การลบข้อมูล Session.....	397

บทที่ 18 Node: การใช้งานร่วมกับฐานข้อมูล 403

บททวนการใช้ MongoDB และ Mongoose.....	403
• การติดตั้ง MongoDB	404
• การเชื่อมต่อกับ MongoDB ด้วย Mongoose.....	405
• การสร้าง Schema และ Model ในเบื้องต้น.....	406
จัดเตรียมการสำหรับเชื่อมโยงส่วนต่างๆ.....	408
การเพิ่มข้อมูล.....	411

• วิธีการทางฝั่ง Local.....	411
• วิธีการทางฝั่ง Server.....	413
การอ่านข้อมูล.....	414
• วิธีการทางฝั่ง Server.....	414
• วิธีการทางฝั่ง Local.....	415
การแก้ไขข้อมูล.....	417
• วิธีการทางฝั่ง Local.....	417
• วิธีการทางฝั่ง Server.....	421
การลบข้อมูล.....	422
• วิธีการทางฝั่ง Local.....	422
• วิธีการทางฝั่ง Server.....	425
เทคนิคการแบ่งเพจ.....	425
• ไลบรารี mongoose-paginate-v2.....	425
• วิธีการทางฝั่ง Server.....	427
• วิธีการทางฝั่ง Local.....	427
Workshop: การค้นหาข้อมูล.....	431

SE-ED

* * inspiration starts here

VS Code และพื้นฐาน HTML5

1

HTML (Hypertext Markup Language) เป็นภาษาสำหรับกำหนดโครงสร้างของเอกสารเว็บเพจในรูปแบบแท็ก (Tag) เช่น `<html>`, `<head>`, `<body>` เป็นต้น ถึงแม้ในปัจจุบันจะมีเทคโนโลยีเกี่ยวกับเว็บออกมาอย่างมากมาย แต่เรายังต้องใช้ HTML ในการจัดวางโครงสร้างของเอกสารเหมือนเดิม โดยเนื้อหาในบทนี้ นอกจากเราจะได้ศึกษาเกี่ยวกับแท็กพื้นฐานที่สำคัญของ HTML แล้ว ยังจะกล่าวถึง Visual Studio Code ซึ่งเป็นเครื่องมือสำหรับการเขียนโค้ดและทดสอบเว็บเพจอีกด้วย

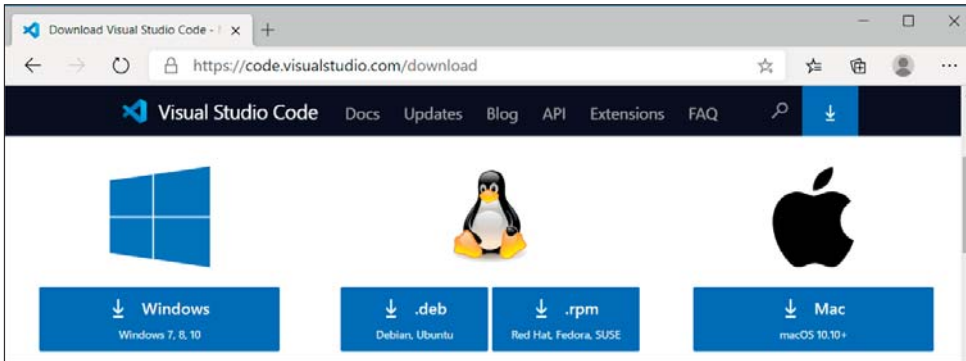
inspiration starts here

การติดตั้งและใช้งาน VS Code

การเขียนโค้ดของเว็บเพจ เช่น HTML5 รวมถึง CSS3 นั้น เราสามารถใช้โปรแกรมประเภท Text Editor ตัวใดก็ได้ สำหรับในหนังสือเล่มนี้ ผู้เขียนเลือกใช้ Visual Studio Code (ต่อไปจะเรียกสั้น ๆ ว่า VS Code) ด้วยเหตุผลดังนี้

- เป็นโปรแกรมประเภท IDE (Integrated Development Environment) สำหรับการเขียนโค้ดทั่วไปที่มีขนาดไม่ใหญ่มาก และได้รับความนิยมสูงสุดในปัจจุบัน
- รองรับการเขียนโค้ดได้หลากหลายภาษา เช่น HTML, JavaScript, Java, PHP, Python เป็นต้น แต่อย่างไรก็ตาม เราอาจต้องติดตั้งส่วนเสริมและตัวแปลภาษานั้น ๆ เพิ่มเติมเข้ามาอีก ยกเว้น HTML เพราะต้องเปิดแสดงผลด้วยเว็บเบราว์เซอร์
- ทำงานได้อย่างชาญฉลาด พร้อมตัวช่วยในการเขียนโค้ดอย่างครบครัน รวมถึงยืดหยุ่นต่อการปรับแต่งให้สามารถใช้ร่วมกับภาษาคอมไพเลอร์อื่นๆ ได้เป็นอย่างดี
- มีส่วนเสริมการทำงานให้เลือกใช้มากมาย และสามารถนำมาใช้งานได้ฟรี

สำหรับชุดติดตั้ง VS Code สามารถดาวน์โหลดได้จากเว็บไซต์ <https://code.visualstudio.com/download> หลังจากดาวน์โหลดเสร็จ ก็ให้ดับเบิลคลิกไฟล์ติดตั้ง ซึ่งขั้นตอนต่างๆ ก็ทำเหมือนกับโปรแกรมทั่วไป

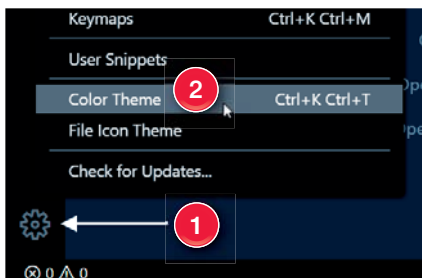


VS Code เป็น IDE สำหรับการเขียนโค้ดทั่วไปที่ไม่เจาะจงว่าเราต้องใช้ภาษาใด ดังนั้น เครื่องมือและตัวช่วยต่างๆ ก็เป็นลักษณะพื้นฐานที่สามารถนำไปใช้ได้กับภาษาคอมพิวเตอร์เกือบทั้งหมด โดยลักษณะที่น่าสนใจซึ่งบางครั้งเราอาจนำไปใช้งานได้ มีดังนี้

การเปลี่ยนรูปแบบสีของ VS Code

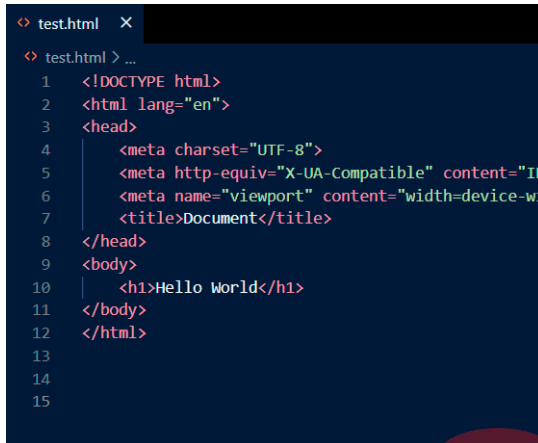
รูปแบบสีหรือ Color Theme ที่ VS Code เลือกไว้ล่วงหน้า จะเป็นสีดำ (Dark) แต่หากเราไม่ชอบสีนี้ VS Code ก็มีรูปแบบสีมาให้เลือกบางส่วน ซึ่งขั้นตอนคือ

1. ที่มุมล่างซ้าย คลิกที่ไอคอน Manage
2. เลือกเมนู Color Theme แล้วเลือกรูปแบบสีที่ต้องการ



การย่อและขยายขนาด (Zoom Out & Zoom In)

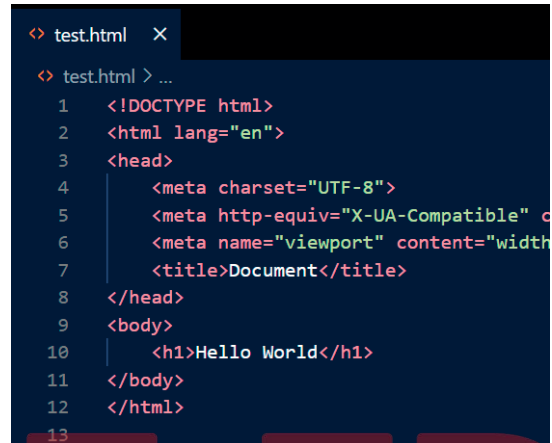
ขนาดปกติของฟอนต์และไอคอนที่ VS Code กำหนดไว้ล่วงหน้า อาจค่อนข้างเล็กสำหรับบางคน ดังภาพต่อไปนี้ (ซ้าย)



```

1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <meta http-equiv="X-UA-Compatible" content="IE=edge">
6   <meta name="viewport" content="width=device-width, initial-scale=1">
7   <title>Document</title>
8 </head>
9 <body>
10  <h1>Hello World</h1>
11 </body>
12 </html>
13
14
15

```



```

1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <meta http-equiv="X-UA-Compatible" content="IE=edge">
6   <meta name="viewport" content="width=device-width, initial-scale=1">
7   <title>Document</title>
8 </head>
9 <body>
10  <h1>Hello World</h1>
11 </body>
12 </html>
13

```

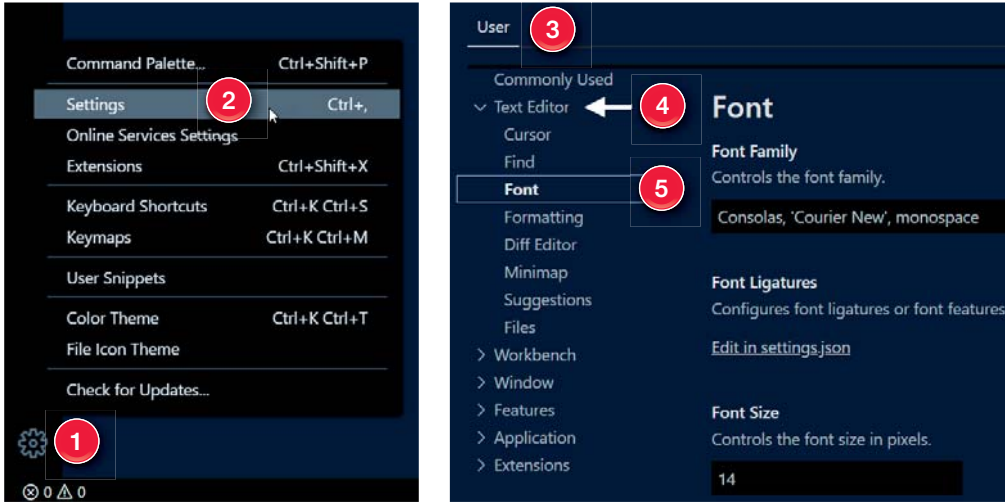
- หากเราต้องการให้ฟอนต์และไอคอนมีขนาดใหญ่ขึ้นอีก 1 ระดับ ก็เลือกที่เมนู **View > Appearance > Zoom In** หรือกด <Ctrl => และถ้าทำเช่นนี้อีก ขนาดจะเพิ่มขึ้นอีก 1 ระดับ ดังภาพบนด้านขวามือ
- หากเราต้องการให้ขนาดของฟอนต์และไอคอนเล็กลง 1 ระดับ ก็เลือกที่เมนู **View > Appearance > Zoom Out** หรือกด <Ctrl -> และถ้าทำเช่นนี้อีก ขนาดจะลดลงอีก 1 ระดับ

การกำหนดรูปแบบฟอนต์ของ Code Editor

หากเราต้องการปรับเปลี่ยนฟอนต์ของ Code Editor ก็มีขั้นตอนดังนี้

1. คลิกที่ไอคอน Manage ตรงมุมล่างซ้าย
2. เลือกที่เมนู Settings
3. เลือกแท็บ User
4. คลิกที่ Text Editor
5. คลิกเมนู Font

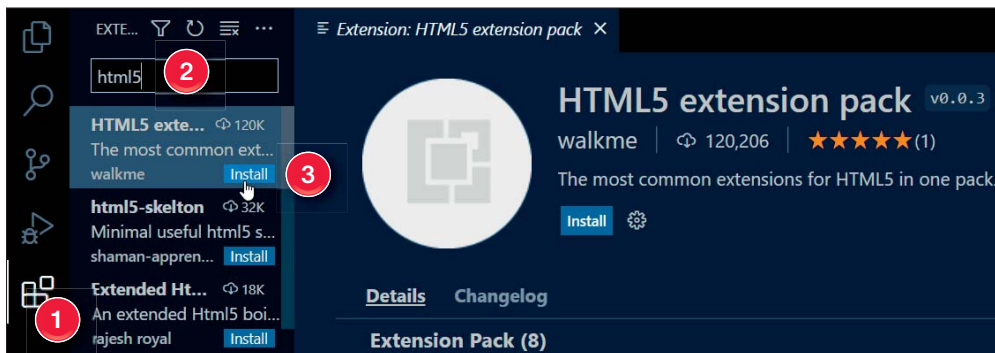
จากนั้นก็กำหนดรูปแบบฟอนต์ตามที่เราต้องการ เช่น Font Family (ชนิดฟอนต์) Font Size และ Font Weight เป็นต้น



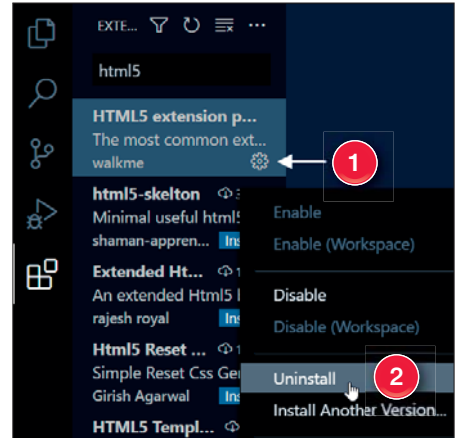
การติดตั้งส่วนขยายของ VS Code

การเขียนโค้ดในบางกรณี เราอาจต้องใช้ส่วนขยาย (Extension) สำหรับการทำงานร่วมกับภาษานั้นๆ ซึ่งเราสามารถตรวจสอบและดาวน์โหลดส่วนขยายมาใช้งานได้ดังนี้

1. ในขณะนั้น ต้องเชื่อมต่อกับอินเทอร์เน็ต แล้วคลิกที่ไอคอน Extensions ที่ขอบด้านซ้าย (ดูภาพถัดไป) หรือเลือกที่เมนู **View > Extensions**
2. ใส่คีย์เวิร์ดลงไป หลังจากนั้นจะปรากฏรายชื่อส่วนขยายที่อาจเกี่ยวข้องกับคีย์เวิร์ดดังกล่าว
3. ถ้าเราต้องการติดตั้งส่วนขยายใด ก็คลิกปุ่ม Install บนรายการนั้น หรือหากต้องการดูรายละเอียดเพิ่มเติมเกี่ยวกับส่วนขยายใด ก็คลิกที่รายการดังกล่าว ดังภาพ



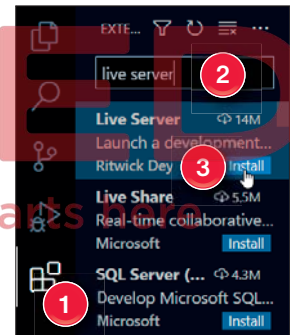
4. ถ้าค้นหาส่วนขยายที่ถูกติดตั้งเอาไว้แล้ว จะมีรูปไอคอน Manage ที่มุมล่างขวาของรายการนั้น ทั้งนี้ หากเราต้องการลบส่วนขยายนั้นทิ้งไป ก็ทำดังนี้
(1) คลิกที่รูปไอคอนดังกล่าว (2) เลือกที่ Uninstall



การทดสอบเว็บเพจด้วย Live Server

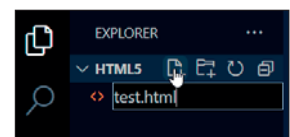
เนื่องจากในหัวข้อต่อๆ ไปของบทนี้ เราต้องเปิดดูผลลัพธ์ด้วยเว็บเบราว์เซอร์ ซึ่งโปรแกรม VS Code นั้นไม่มีเครื่องมือที่รองรับการทดสอบเช่นนี้โดยตรง แต่เราสามารถติดตั้งส่วนเสริมของ VS Code สำหรับสร้างเซิร์ฟเวอร์ (Server) แบบจำลอง เพื่อเปิดไฟล์เว็บเพจด้วยเบราว์เซอร์โดยตรงได้ ดังนั้น การตรวจสอบผลลัพธ์จึงสะดวกรวดเร็วยิ่งขึ้น ซึ่งมีขั้นตอนดังนี้

1. เปิดเข้าสู่ VS Code แล้วคลิกที่ไอคอน Extensions ที่แถบด้านซ้ายมือ (ต้องเชื่อมต่ออินเทอร์เน็ต)
2. ในช่องค้นหา ให้พิมพ์คำว่า Live Server ลงไป
3. หลังจากปรากฏรายชื่อส่วนเสริม Live Server ให้คลิกปุ่ม Install

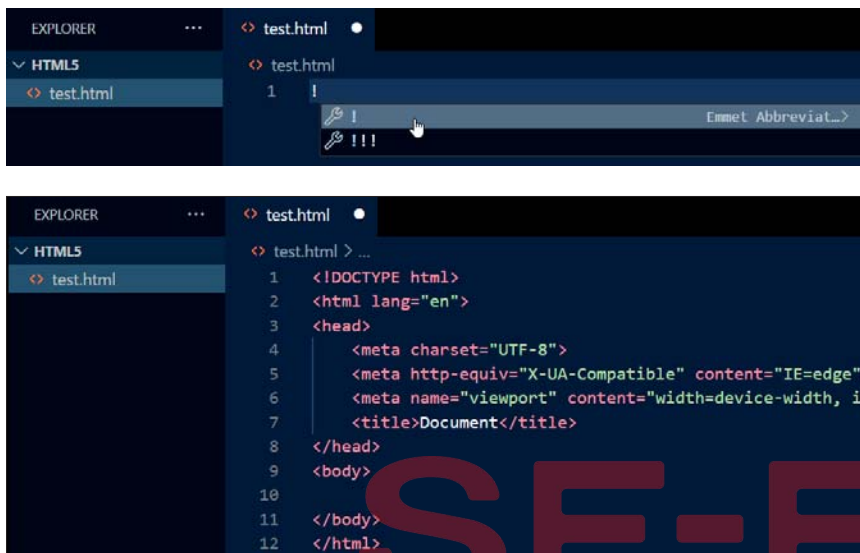


หลังการติดตั้ง ให้ลองทดสอบการทำงานของ Live Server ดังขั้นตอนต่อไปนี้

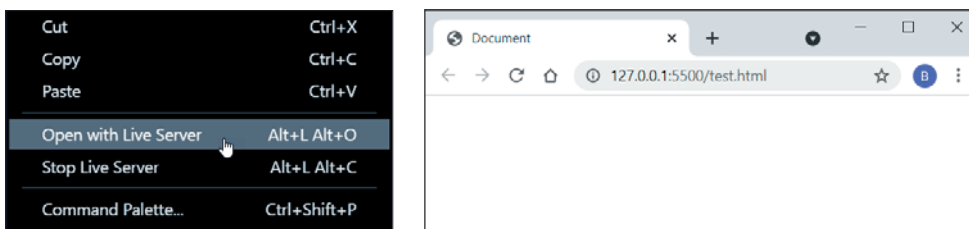
- ข้อบังคับของ Live Server คือ เราต้องมีโฟลเดอร์ที่แน่นอนในการจัดเก็บไฟล์เว็บเพจ โดยในที่นี้ผู้เขียนจะเก็บไฟล์สำหรับทดสอบไว้ที่ C:\bvn\html5 ดังนั้น ให้เราสร้างโฟลเดอร์ว่างๆ ตามนี้ขึ้นมาก่อน จะด้วยวิธีใดก็ได้
- ที่ VS Code ให้เปิดโฟลเดอร์โดยเลือกเมนู File > Open Folder จากนั้นเลือกโฟลเดอร์ที่เราได้สร้างเอาไว้ ซึ่งในที่นี้คือ C:\bvn\html5
- เมื่อเปิดโฟลเดอร์แล้ว ให้สร้างไฟล์ใหม่เพิ่มเข้ามาในโฟลเดอร์นั้น โดยคลิกไอคอนดังในภาพ แล้วกำหนดชื่อไฟล์ลงไป โดยให้มีส่วนขยายเป็น .html เช่น ในที่กำหนดกำหนดชื่อเป็น test.html (ตำแหน่งไฟล์คือ C:\bvn\html5\test.html)



- ให้เราใช้ตัวช่วยคือ Emmet Action ในการสร้างเอกสาร โดยพิมพ์เครื่องหมาย ! ลงบน Editor เมื่อปรากฏ IntelliSense ให้เลื่อนไฮไลต์มาที่ ! Emmet Abbreviation แล้วกด <Enter> หรือคลิกด้วยเมาส์ ซึ่งโครงสร้างหลักของ HTML จะถูกใส่ลงไปโดยอัตโนมัติ



- ให้บันทึกการเปลี่ยนแปลง เช่น กด <Ctrl + S> แล้วคลิกขวาบน Editor ที่แสดงโค้ดของไฟล์ หรือคลิกขวาที่ชื่อไฟล์ก็ได้ จากนั้นเลือก **Open with Live Server** แล้วไฟล์ของเว็บเพจดังกล่าว จะถูกเปิดบนเว็บเบราว์เซอร์ที่ตั้งค่าเป็นดีฟอลต์ให้โดยอัตโนมัติ



ต่อไปเมื่อเราแก้ไขโค้ดแล้วบันทึกการเปลี่ยนแปลง Live Server จะอัปเดตการแสดงผลบนเว็บเบราว์เซอร์ให้สอดคล้องกันทันที ซึ่งเราไม่จำเป็นต้อง Refresh เบราว์เซอร์ก็ได้

หมายเหตุ

อย่าลืมข้อบังคับของ Live Server นั่นคือ ต้องเปิดโฟลเดอร์ที่จะเก็บไฟล์ขึ้นมาก่อน แล้วค่อยเปิดไฟล์เป้าหมาย จากนั้นจึงแสดงผลด้วยเบราว์เซอร์ ตามขั้นตอนที่ได้กล่าวมาแล้วนั่นเอง หากเราเปิดไฟล์บน VS Code โดยตรง จะไม่สามารถแสดงด้วยเบราว์เซอร์ได้

ลักษณะของแท็กและอิลิเมนต์

องค์ประกอบต่างๆ ภายในเว็บเพจ จะถูกกำหนดด้วยสัญลักษณ์ที่เรียกว่า แท็ก (Tag) เพื่อแสดงจุดเริ่มต้นและจุดสิ้นสุด (หรือขอบเขต) ของเนื้อหาในแต่ละอย่าง ดังรูปแบบต่อไปนี้



เราสามารถสรุปลักษณะพื้นฐานที่สำคัญเกี่ยวกับแท็กของ HTML ได้ดังนี้

- ชื่อแท็ก จะกำหนดระหว่างเครื่องหมาย < และ > เช่น <html> หรือ <title> เป็นต้น
- แท็กที่กำหนดไว้ที่จุดเริ่มต้นของเนื้อหาจะเรียกว่า Start Tag หรือ Open Tag
- แท็กที่กำหนดไว้ที่จุดสิ้นสุดของเนื้อหาจะเรียกว่า End Tag หรือ Close Tag ซึ่งจะเขียนเครื่องหมาย / ไว้หน้าชื่อแท็กด้วย เช่น </html>
- เมื่อรวมแท็กและเนื้อหาเข้าด้วยกัน เราจะเรียกว่า อิลิเมนต์ (Element) ดังนั้น จึงกล่าวได้ว่า อิลิเมนต์ ก็คือสิ่งที่อยู่ระหว่าง Start Tag และ End Tag นั่นเอง
- คู่แท็กเริ่มต้นและสิ้นสุดต้องเป็นชื่ออันเดียวกัน เช่น <body> กับ </body> หรือ <title> กับ </title> เป็นต้น
- แท็กของ HTML ไม่สนใจความแตกต่างของตัวพิมพ์ เช่น เราอาจเขียนเป็น <title> หรือ <TITLE> หรือใช้ตัวพิมพ์เล็ก-ใหญ่ปะปนอย่างไรก็ได้ แต่โดยทั่วไปนิยมเขียนแท็กด้วยตัวพิมพ์แบบเดียวกันทั้งหมด
- ยังมีอิลิเมนต์อีกลักษณะหนึ่งที่ไม่ต้องกำหนดเนื้อหาให้กับมัน โดยจะเขียนด้วยแท็กเริ่มต้นเพียงอันเดียว ซึ่งอาจเรียกว่าเป็นอิลิเมนต์แบบแท็กเดี่ยว (Single Tag) เช่น
, <hr> เป็นต้น

แอตทริบิวต์ของแท็ก

แอตทริบิวต์ (Attribute) หรือพรีอเพอร์ติ (Property) เป็นการกำหนดข้อมูลบางอย่างให้กับแท็กต่างๆ เช่น แท็กที่ใช้แสดงรูปภาพ ก็จะมีแอตทริบิวต์ที่ใช้กำหนดตำแหน่งภาพ หรือแท็กที่แสดงลิงก์ ก็จะมีแอตทริบิวต์ที่ใช้กำหนด URL ปลายทางเมื่อคลิกที่ลิงก์ เป็นต้น ซึ่งการกำหนดแอตทริบิวต์มี 2 ลักษณะ ดังนี้

- **<name="value">** แอตทริบิวต์แบบนี้จะกำหนดชื่อและค่าของมันในรูปแบบ name="value" ที่แท็กเริ่มต้น ในรูปแบบ

```
<ชื่อแท็ก attribute1="value" attribute2="value" attribute3="value" ...>
```

```

<a href="http://www.developerthai.com" target="_blank">Go to...</a>
```

- **<attribute_name>** เรียกว่า แอตทริบิวต์แบบบูลีน (boolean) โดยทั่วไปเราจะกำหนดเพียงแค่ชื่อแอตทริบิวต์ เพื่อบ่งชี้ว่าเราต้องการใช้งานแอตทริบิวต์ดังกล่าวโดยไม่ระบุค่า ซึ่งแอตทริบิวต์ในกลุ่มนี้ เช่น disabled, readonly, checked, selected, multiple เป็นต้น แนวทางการใช้งานคือ

```
<input type="text" disabled>
<input type="checkbox" checked>
```

inspiration starts here

โครงสร้างหลักของ HTML

โครงสร้างหลักของเว็บเพจ จะกำหนดด้วยแท็กต่างๆ ดังต่อไปนี้

<!doctype html>	ประกาศว่าเป็นเอกสารชนิด html
<html>	กำหนดองค์ประกอบภายในเว็บเพจ นอกจากนี้ เราอาจจะระบุแอตทริบิวต์ lang เพื่อบ่งชี้ภาษาที่ใช้ในการแสดงผล เช่น <html lang="th">
<head>	กำหนดองค์ประกอบในส่วนหัวของเว็บเพจ
<body>	กำหนดส่วนเนื้อหาของเว็บเพจที่จะแสดงผล

ลักษณะการจัดวางแท็กเพื่อกำหนดโครงสร้างหลักของเพจ จะเป็นดังนี้

```
<!doctype html>
<html lang="th">
<head>
    ส่วนหัวของเว็บเพจ (ดูแท็กกลุ่มนี้ในหัวข้อถัดไป)
</head>
<body>
    เนื้อหาของเว็บเพจ
</body>
</html>
```

กลุ่มแท็กที่ส่วนหัวของเว็บเพจ

ส่วนหัวของเว็บเพจ จะกำหนดด้วยแท็ก <head> แต่เราจะไม่ระบุแอตทริบิวต์ให้กับมันโดยตรง เพราะมีแท็กที่ใช้งานเฉพาะในแต่ละอย่างอยู่แล้ว โดยวางซ้อนไว้ใน <head> แต่แท็กเหล่านี้จะไม่แสดงผลออกไปที่เบราว์เซอร์ ซึ่งอันที่น่าสนใจคือ

<title>	กำหนดหัวข้อหรือชื่อเว็บเพจ ซึ่งจะปรากฏบนแท็บของเบราว์เซอร์
<meta charset="...">	กำหนดรหัสอักขระสำหรับเพจนั้น เช่น <meta charset="UTF-8">
<meta name="viewport">	กำหนดขนาดความกว้างของพื้นที่แสดงผล
<meta name="keywords">	กำหนดคีย์เวิร์ดที่มีในเพจนั้น เพื่อให้ Search Engine นำไปทำเป็นดัชนีการสืบค้นข้อมูล โดยใช้แอตทริบิวต์ content สำหรับระบุคีย์เวิร์ด หากมีมากกว่า 1 คำ ให้คั่นแต่ละคำด้วยเครื่องหมาย , เช่น <meta name="keywords" content="Bootstrap5,Vue,Node">
<meta name="description">	สำหรับเขียนคำอธิบายโดยสังเขปเกี่ยวกับเพจนั้น เพื่อให้ Search Engine นำไปแสดงในผลลัพธ์ของการสืบค้น โดยให้เขียนคำอธิบายด้วยแอตทริบิวต์ content เช่น <meta name="description" content="HTML5 is Hyper ...">
<meta http-equiv="...">	สำหรับกำหนด Http Header ของเว็บเพจ เพื่อให้เบราว์เซอร์นำไปพิจารณารูปแบบการตอบสนอง
<base href="...">	กำหนดส่วนเริ่มต้นของ URL ที่มีอ้างอิงบ่อยๆ ในเพจนั้น

นอกจากนี้ ยังมีแท็กอีกบางส่วนที่ต้องกำหนดไว้ที่ส่วนหัวของเว็บเพจ เช่น แท็กเกี่ยวกับ CSS หรือ JavaScript เป็นต้น ซึ่งจะนำไปกล่าวถึงในเรื่องที่เกี่ยวข้องต่อไป และสำหรับบางแท็กที่แสดงในตารางที่ผ่านมา มีรายละเอียดเพิ่มเติม ดังนี้

- **viewport** เป็นขนาดพื้นที่ในการแสดงผลเว็บเพจ เนื่องจากขนาดหน้าจอของอุปกรณ์ (Device) ที่ใช้เปิดดูเว็บอาจแตกต่างกันออกไป เช่น ในคอมพิวเตอร์จะมีขนาดใหญ่กว่ามือถือ เป็นต้น ดังนั้น เราสามารถกำหนด viewport ให้ขึ้นกับอุปกรณ์ เช่น

```
<meta name="viewport" content="width=device-width, initial-scale=1.0">
```

- **width=device-width** หมายถึง ให้พื้นที่การแสดงผลขึ้นกับขนาดความกว้างของอุปกรณ์
- **initial-scale=1.0** หมายถึง ให้อัตราส่วนการขยายขนาด (zoom) เท่ากับ 1.0
- **http-equiv** ลักษณะการใช้งานที่เราพบเห็นได้บ่อยๆ คือ
 - ให้เบราว์เซอร์โหลดเพจนั้นมาแสดงใหม่ในทุกช่วงเวลาที่กำหนด (วินาที) เช่น ในโค้ดถัดไป หมายถึง ให้โหลดเพจทุก 30 วินาที

```
<meta http-equiv="refresh" content="30">
```

- หรืออาจให้เบราว์เซอร์ย้ายไปแสดงผลที่เพจหรือ URL ที่ระบุ เมื่อครบเวลา (วินาที) ที่กำหนดหลังจากโหลดเพจนั้น เช่น ในโค้ดถัดไปหมายถึง หลังจากโหลดเพจนี้แล้ว ให้จับเวลาไปอีก 5 วินาที แล้วย้ายไปยัง <http://www.example.com>

```
<meta http-equiv="refresh" content="5;http://www.example.com">
```

- นอกจากนี้ ยังใช้กำหนดเวอร์ชันของเว็บเบราว์เซอร์ที่จะยึดเป็นเกณฑ์ในการแสดงผล เช่น ในโค้ดถัดไป หมายถึง ถ้าเว็บเพจนี้ถูกเปิดด้วย IE ให้แสดงผลตามข้อกำหนดของเวอร์ชันล่าสุดในขณะนั้น เป็นต้น

```
<meta http-equiv="X-UA-Compatible" content="IE=edge">
```

- **<base>** มักใช้เมื่อเราต้องเขียน URL ที่ยาวๆ อยู่บ่อยๆ ซึ่งหากเรานำส่วนเริ่มต้นมากำหนดด้วยแท็ก **<base>** แล้วต่อไป ถ้าต้องการอ้างถึง URL ดังกล่าว ก็สามารถกำหนดเฉพาะส่วนที่อยู่ต่อจาก base ได้เลย อย่างไรก็ตาม ในแต่ละเพจ จะมีแท็ก **<base>** ได้เพียงอันเดียวเท่านั้น เช่น สมมติว่าโดยปกติเราต้องกำหนดตำแหน่ง URL ของหลายๆ ภาพและลิงก์ซ้ำกันในลักษณะดังนี้

```


```

เราสามารถนำ URL ส่วนที่ต้องอ้างอิงซ้ำๆ กัน ไปกำหนดด้วยแท็ก <base> ไว้ที่ส่วนหัว (head) ของเพจ แล้วต่อไปก็สามารถอ้างอิง URL เฉพาะส่วนที่อยู่ต่อจาก base ได้เลย แต่ URL ที่ไม่ได้อยู่ที่เดียวกับ base ต้องระบุแบบเต็มๆ ตามปกติ เช่น

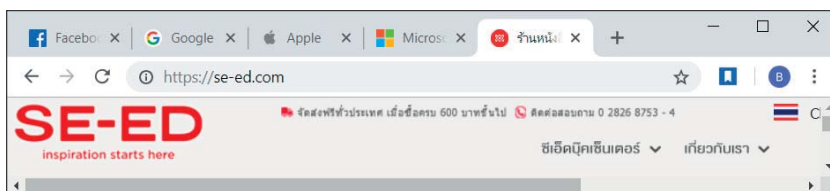
```
...
<head>
  <base href="http://www.developerthai.com/img/">
</head>
<body>
  
  
</body>
```

- แท็ก <base> ยังมีอีกหนึ่งแอตทริบิวต์คือ target ซึ่งมักใช้ในกรณีที่เราต้องการให้เปิดเพจเป้าหมายบนแท็บใหม่ โดยกำหนดค่าแอตทริบิวต์นี้เป็น "_blank" เช่น
 <base href="http://www.example.com" target="_blank">

inspiration starts here

การกำหนดภาพ Favicon

Favicon (มาจากคำว่า Favorite icon) เป็นภาพไอคอนขนาดเล็ก ซึ่งมักใช้เพื่อแสดงความเป็นเอกลักษณ์ของเว็บไซต์แห่งนั้น เช่น ภาพโลโก้ เป็นต้น โดยจะถูกนำมาแสดงที่แท็บของเบราว์เซอร์ร่วมกับไตเติลของเพจนั้น ในลักษณะดังภาพต่อไปนี้



สำหรับแนวทางการสร้างภาพ Favicon มีดังนี้

- สำหรับภาพ Favicon ส่วนใหญ่นิยมกำหนดให้มีขนาด 16×16 ปัจจุบันเราสามารถใช้ภาพได้หลายชนิด เช่น gif, png, svg เป็นต้น
- วิธีการนำภาพ Favicon มาใช้งานคือ ให้กำหนดผ่านแท็ก `<link>` ในรูปแบบ

```
<link rel="icon" href="ชื่อภาพ" type="...">
```

- แท็ก `<link>` ต้องวางไว้ระหว่าง `<head>...</head>` เช่น

```
<head>
...
<link rel="icon" href="favicon.gif" type="image/gif">
<!--
<link rel="icon" href="favicon.svg" type="image/svg+xml">
-->
</head>
```

กลุ่มแท็กสำหรับจัดแบ่งเนื้อหา

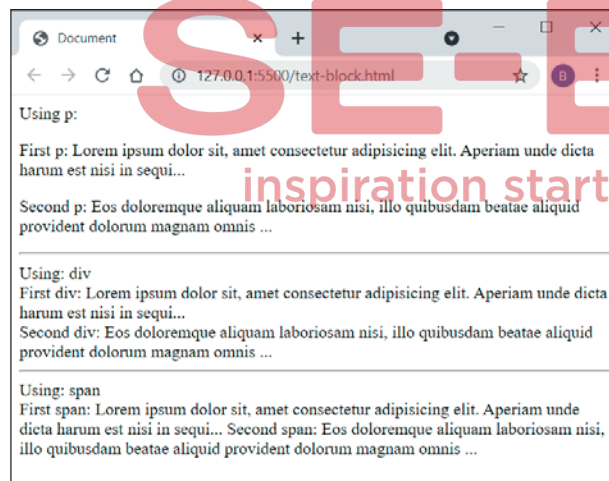
ในกรณีที่เราต้องการจัดแบ่งขอบเขตของเนื้อหา เช่น การขึ้นบรรทัดใหม่ การขึ้นย่อหน้า หรือการแบ่งกลุ่มข้อความเพื่อการจัดรูปแบบเฉพาะ ก็สามารถกำหนดด้วยแท็กต่อไปนี้

<code>
</code>	ใช้กำหนดจุดสิ้นสุด (Break) เพื่อขึ้นบรรทัดใหม่ แท็กนี้เป็นแบบ Single Tag
<code><p></code>	สำหรับการขึ้นย่อหน้าใหม่ โดยกำหนดข้อความไว้ระหว่างแท็กเปิด <code><p></code> และแท็กปิด <code></p></code> แล้วข้อความนั้นจะถูกนำไปขึ้นบรรทัดใหม่ โดยเว้นระยะห่างพอประมาณ
<code></code>	ใช้จัดกลุ่มเนื้อหาภายในบรรทัด (inline) โดยเนื้อหาที่อยู่ใน <code></code> สามารถรวมอยู่ในบรรทัดเดียวกันกับเนื้อหาที่อยู่นอก <code></code> ได้
<code><div></code>	ใช้จัดกลุ่มเนื้อหาแบบบล็อก (block) โดยเนื้อหาที่อยู่ใน <code><div></code> ไม่สามารถรวมอยู่ในบรรทัดเดียวกันกับเนื้อหาที่นอก <code><div></code> ได้ แต่จะถูกแยกออกไปอยู่บรรทัดใหม่
<code><hr></code>	สำหรับแสดงเส้นแบ่งในแนวนอน (Horizontal Rule) แท็กนี้เป็นแบบ Single Tag

แนวทางการใช้งาน เช่น

bvn/html5/text-block.html

```
...
<body>
  <span>Using p:</span><br>
  <p>First p: Lorem ipsum dolor sit amet consectetur ...</p>
  <p>Second p: Eos doloremque aliquam laboriosam nisi ...</p>
  <hr>
  <span>Using: div</span><br>
  <div>First div: Lorem ipsum dolor sit amet consectetur ...</div>
  <div>Second div: Eos doloremque aliquam laboriosam nisi ...</div>
  <hr>
  <span>Using: span</span><br>
  <span>First span: Lorem ipsum dolor sit amet consectetur ...</span>
  <span>Second span: Eos doloremque aliquam laboriosam nisi ...</span>
</body>
```



จากภาพผลลัพธ์จะเห็นว่า กรณีของ `<p>` จะแบ่งย่อหน้าโดยขึ้นบรรทัดใหม่พร้อมเว้นบรรทัดว่างระยะระหว่างย่อหน้า ส่วน `<div>` จะขึ้นบรรทัดใหม่แต่ไม่เว้นบรรทัดว่าง ส่วน `<span>` จะเขียนต่อเนื่องในบรรทัดเดียวกัน

กลุ่มแท็กสำหรับจัดรูปแบบข้อความ

การจัดรูปแบบข้อความ เช่น ทำตัวหนา ตัวเอียง ชิดเส้นใต้ เป็นต้น โดยมีแท็กที่น่าสนใจ ซึ่งเรามักนำไปใช้งานอยู่บ่อยๆ คือ

	สำหรับทำตัวหนา (bold)
<h1>, <h2>, ... <h6>	กำหนดขนาดหัวข้อโดย <h1> จะใหญ่สุดแล้วลดลงตามลำดับ
<i>	สำหรับทำตัวเอียง (italic)
<u>	สำหรับชิดเส้นใต้ (underline)
<s> หรือ 	สำหรับชิดเส้นทับข้อความ
<mark>	ทำแถบสีพื้นหลังข้อความ (ปกติจะเป็นสีเหลือง)
<sub>	สำหรับทำตัวห้อย (subscript) เช่น O ₂
<sup>	สำหรับทำตัวยก (superscript) เช่น e = mc ²
<pre>	จัดรูปแบบข้อความตามที่เขียนในโค้ด (pre-format)

แนวทางการใช้งาน เช่น

bvn/html5/text-format.html

```

...
<body>
  <p><b>ข้อความนี้เป็นตัวหนา (bold)</b></p>
  <p><i>ข้อความนี้เป็นตัวเอียง (italic)</i></p>
  <p><u>ข้อความนี้ชิดเส้นใต้ (underline)</u></p>
  <p><del>ข้อความนี้ถูกขีดทับ (delete)</del></p>
  <p>ข้อความนี้มี<sub>ตัวห้อย</sub> (subscript)</p>
  <p>ข้อความนี้มี<sup>ตัวยก</sup> (superscript)</p>
  <p><mark>ข้อความนี้ทำแถบสี (mark)</mark></p>
  <h1>ข้อความขนาด h1</h1>
  <h3>ข้อความขนาด h3</h3>
  <h5>ข้อความขนาด h5</h5>
  <pre>
    บรรทัดที่ 1
    บรรทัดที่ 2
    บรรทัดที่ 3
  </pre>
</body>

```

แท็กสำหรับแสดงคำอธิบาย

กรณีที่เรต้องการเขียนคำอธิบาย (Comment) ลงในโค้ด ให้กำหนดด้วยแท็ก `<!-- ... -->` โดยเว็บเบราว์เซอร์จะไม่นำคำอธิบายนี้ไปประมวลผล และไม่แสดงออกมาที่หน้าเว็บเพจให้เห็น หรืออาจเขียนครอบคำสั่ง HTML เพื่อไม่ให้เว็บเบราว์เซอร์ประมวลคำสั่งนั้นก็ได้ เช่น

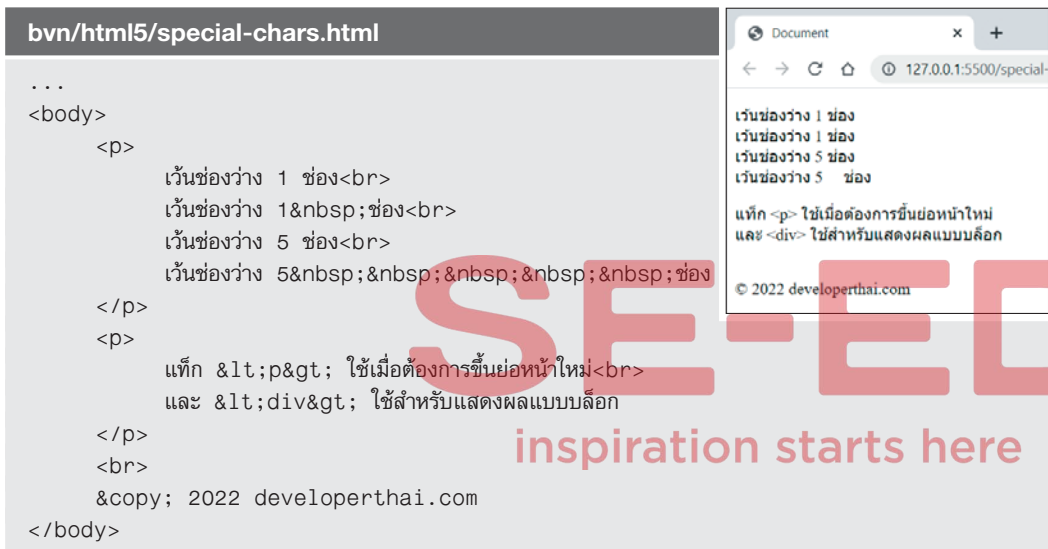
```
<body>
  <!-- เริ่มต้นบทความ -->
  <div>
    Lorem ipsum dolor sit amet, <br>
    consectetur adipiscing elit,
    ...
  </div>
  <!-- ล้นสุดบทความ -->
  <!-- &copy;2022 developerthai.com -->
</body>
```

อักขระพิเศษของ HTML

อักขระบางตัว ถูกนำมาใช้สำหรับการกำหนดแท็ก HTML เช่น เครื่องหมาย `<` หรือ `>` และนอกจากนี้ อักขระอีกบางส่วนก็ไม่มีอยู่บนคีย์บอร์ด จึงไม่สามารถเขียนอักขระลงไปโดยตรงได้ แต่ต้องแทนด้วยสัญลักษณ์พิเศษ ซึ่งอันที่น่าสนใจคือ

ช่องว่าง	 (แทนช่องว่าง 1 ช่อง)	x	× (คูณ)
<	< (มาจาก less than)	±	±
>	> (มาจาก greater than)	·	· (จุดกลางบรรทัด)
"	";	£	£
'	';	€	€
&	&;	¥	¥
©	© (มาจาก copyrights)	«	«
®	® (มาจาก registered)	»	»
÷	÷	é	é

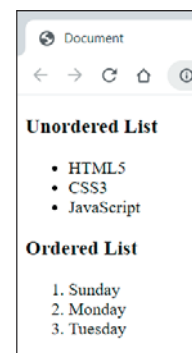
- ถ้าเว้นช่องว่างเพียงอันเดียว เราสามารถใส่ช่องว่างลงไปโค้ดโดยตรงได้เลย แต่ถ้าเว้นมากกว่า 1 ช่อง เราต้องแทนด้วยสัญลักษณ์ ` ` ตามจำนวนช่องว่างที่ต้องการ ทั้งนี้หากเว้นช่องหลายช่องในโค้ดโดยตรง เบราวเซอร์จะแสดงช่องว่างเพียงอันเดียว
- สำหรับอักขระ `>`, `<`, `"`, `'` และ `&` หากใช้ในการเขียนแบบข้อความทั่วไปที่ไม่ใช่แท็ก HTML สามารถเขียนลงไปโดยตรงได้เลย เช่น เลือกสินค้า `>` หยิบใส่รถเข็น `>` ชำระเงิน แต่ถ้าจะนำไปเขียนร่วมกับแท็ก HTML และต้องการให้เบราวเซอร์แสดงอักขระนั้นออกมา จะต้องเขียนในแบบสัญลักษณ์ดังกล่าวมาแล้ว เช่น แท็ก `<bold>` ก็เขียนแทนด้วย `<bold>`



แท็กสำหรับแสดงรายการ

ลำดับรายการ (List) ก็คือการแบ่งข้อมูลเป็นข้อย่อย ๆ ซึ่งเราสามารถกำหนดรูปแบบการจัดเรียงได้ 2 ประเภทคือ

- **Unordered List** เป็นการจัดเรียงรายการที่ไม่เป็นลำดับต่อเนื่อง โดยใช้สัญลักษณ์ เช่น วงกลมหรือสี่เหลี่ยมขนาดเล็ก วางไว้หน้ารายการ
- **Ordered List** เป็นการจัดเรียงรายการที่มีลำดับต่อเนื่องโดยใช้ตัวเลขหรืออักขระ เช่น 1, 2, 3, ... หรือ a, b, c, ... วางไว้หน้ารายการ



สำหรับแท็กที่เกี่ยวข้องกับการกำหนดลำดับรายการ มีดังนี้

<code><ul type="..."></code>	กำหนดขอบเขตของรายการแบบ Unordered List โดยแอตทริบิวต์ type ใช้กำหนดสัญลักษณ์หน้ารายการ ด้วยค่าใดค่าหนึ่งดังนี้ • disc วงกลมทึบ (ค่าดีฟอลต์) • circle วงกลมกลวง • square สี่เหลี่ยม
<code><ol type="..." start="..."></code>	กำหนดขอบเขตของรายการแบบ Ordered List โดยแอตทริบิวต์ type คืออักขระที่แสดงลำดับหน้ารายการ ให้กำหนดค่าใดค่าหนึ่งดังนี้ • "1" เป็นแบบตัวเลข เช่น 1, 2, 3 • "a" เป็นตัวอักษรภาษาอังกฤษแบบพิมพ์เล็ก เช่น a, b, c, ... • "A" เป็นตัวอักษรภาษาอังกฤษแบบพิมพ์ใหญ่ เช่น A, B, C, ... • "i" ตัวเลขโรมันแบบพิมพ์เล็ก เช่น i, ii, iii, ... • "I" ตัวเลขโรมันแบบพิมพ์ใหญ่ เช่น I, II, III, ... ส่วนแอตทริบิวต์ start ใช้กำหนดลำดับเริ่มต้น (ถ้าไม่ระบุจะเริ่มจากตัวแรกสุดของลำดับในกลุ่มนั้น หรือตามค่าของแอตทริบิวต์ type นั้นเอง)
<code></code>	ใช้กำหนดรายการทั้งแบบ Unordered และ Ordered List

inspiration starts here

แนวทางการกำหนดรายการแบบ Unordered List มีดังนี้

bvn/html5/unordered-list.html

```

...
<body>
  <h5>Unordered list type: disc</h5>
  <ul type="disc">
    <li>Apple</li>
    <li>Banana</li>
    <li>Coconut</li>
  </ul>
  <h5>Unordered list type: square</h5>
  <ul type="square">
    <li>Rose</li>
    <li>Orchid</li>
    <li>Jasmine</li>
  </ul>
  <h5>Unordered list type: circle</h5>
  <ul type="circle">

```

Document

127.0.0.1:55

Unordered list type: disc

- Apple
- Banana
- Coconut

Unordered list type: square

- Rose
- Orchid
- Jasmine

Unordered list type: circle

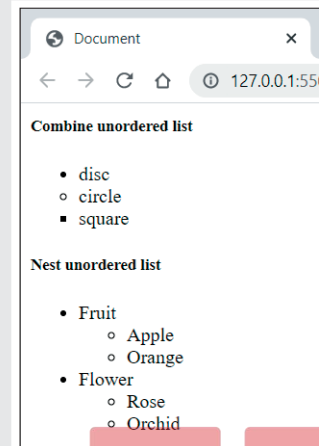
- Red
- Green
- Blue

bvn/html5/unordered-list.html (ต่อ)

```

        <li>Red</li>
        <li>Green</li>
        <li>Blue</li>
    </ul>
    <h5>Combine unordered list</h5>
    <ul type="disc">
        <li>disc</li>
        <li type="circle">circle</li>
        <li type="square">square</li>
    </ul>
    <h5>Nest unordered list</h5>
    <ul>
        <li>Fruit</li>
        <ul>
            <li>Apple</li>
            <li>Orange</li>
        </ul>
        <li>Flower</li>
        <ul>
            <li>Rose</li>
            <li>Orchid</li>
        </ul>
    </ul>
</body>

```



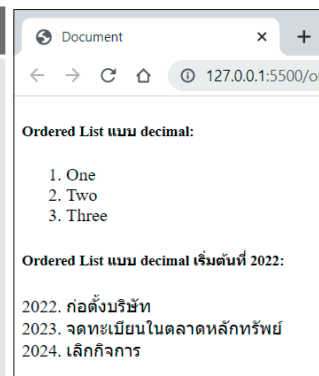
แนวทางการกำหนดรายการแบบ Ordered List มีดังนี้

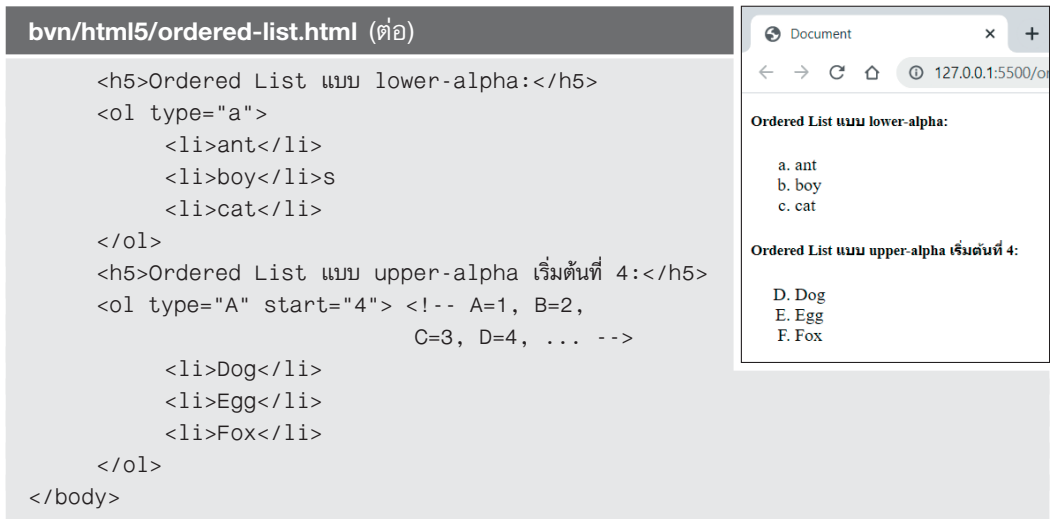
bvn/html5/ordered-list.html

```

...
<body>
    <h5>Ordered List แบบ decimal:</h5>
    <ol type="1">
        <li>One</li>
        <li>Two</li>
        <li>Three</li>
    </ol>
    <h5>Ordered List แบบ decimal เริ่มต้นที่ 2022:</h5>
    <ol type="1" start="2022">
        <li>ก่อตั้งบริษัท</li>
        <li>จดทะเบียนในตลาดหลักทรัพย์</li>
        <li>เลิกกิจการ</li>
    </ol>

```





การเชื่อมโยงด้วยลิงก์

การเชื่อมโยงด้วยลิงก์ (Link) มีหลายลักษณะ เช่น การเชื่อมโยงระหว่างเพจทั้งที่อยู่ในเว็บไซต์เดียวกัน และเว็บไซต์อื่นๆ รวมถึงการเชื่อมโยงสำหรับดาวน์โหลดไฟล์ เป็นต้น โดยจะกำหนดด้วยแท็ก `<a>` ในลักษณะดังนี้

```
<a href="..." target="...">ข้อความที่แสดงเป็นลิงก์</a>
```

โดยการกำหนดแอตทริบิวต์จะขึ้นกับลักษณะการเชื่อมโยง ซึ่งแบ่งเป็นกรณีย่อยๆ คือ

`<a href="url" target="...">`

สำหรับเชื่อมโยงไปเพจอื่นๆ โดยระบุ URL ของเพจปลายทาง เช่น

```
<a href="http://developerthai.com">Go</a>
```

หรือถ้าเพจนั้นอยู่ในเว็บไซต์เดียวกัน ก็อาจระบุแค่ตำแหน่งไฟล์ เช่น

```
<a href="member/signin.html">เข้าสู่ระบบ</a>
```

ส่วนแอตทริบิวต์ `target` ใช้กำหนดวินโดว์เป้าหมายที่จะเปิดลิงก์นั้น ด้วยค่าใดค่าหนึ่งดังนี้

- `_blank` เปิดในวินโดว์หรือแท็บใหม่
- `_self` เปิดในวินโดว์ปัจจุบัน
- `_top` มักใช้ในกรณีที่แสดงเพจในเฟรม แล้วต้องการเปิดเพจนั้นในวินโดว์ปัจจุบันแบบเต็มเพจ หรือยกเลิกเฟรมนั่นเอง

<code></code>	สร้างลิงก์เพื่อส่งอีเมล โดยเมื่อคลิกที่ลิงก์ จะเปิดแอปสำหรับการส่งเมล เช่น <code>Send Mail</code>
<code></code>	สร้างลิงก์สำหรับการดาวน์โหลดไฟล์ ซึ่งหากอยู่บนเว็บไซต์อื่น ต้องระบุ URL แบบเต็ม แต่ถ้าอยู่บนเว็บไซต์ของเราเอง อาจระบุแค่ตำแหน่งไฟล์ก็ได้ เช่น <code>Download</code> <code>Download Code</code>

นอกจากนี้ ยังสามารถสร้างลิงก์เพื่อเลื่อนไปยังหัวข้อที่ต้องการภายในเพจนั้น ซึ่งเรามักจะสร้างลิงก์แบบนี้กับเพจที่มีเนื้อหาค่อนข้างยาว และยุ่งยากต่อการค้นหาข้อมูล โดยการเชื่อมโยงดังกล่าวจะช่วยให้เราเข้าถึงข้อมูลเป้าหมายได้เร็วขึ้น ดังหลักการต่อไปนี้

1. การสร้างลิงก์เพื่อเชื่อมโยงไปยังเป้าหมายที่ได้จัดเตรียมเอาไว้ โดยใช้แท็ก `<a>` เหมือนการสร้างลิงก์ตามปกติ แต่กำหนดค่าแอตทริบิวต์ `href` ในแบบ `<a href="#id">`
2. กำหนดเป้าหมายของการเชื่อมโยง ซึ่งเป็นจุดที่เราจะเลื่อนไปเมื่อคลิกที่ลิงก์ในข้อ 1 โดยวางแท็ก `<a>` พร้อมกำหนดแอตทริบิวต์ `id` เป็นชื่อจุดเชื่อมโยงของตำแหน่งนั้น

bvn/html5/internal-link.html

```
...
<body>
  <a href="#html">HTML</a> -
  <a href="#css">CSS</a> -
  <a href="#js">JavaScript</a>
  <hr><br><br><br><br><br><br>
  <a id="html"></a>
  <h4>HTML</h4>
  <p>HyperText Markup Language(HTML) is the main markup ...</p>
  <a id="css"></a>
  <h4>CSS</h4>
  <p>Cascading Style Sheets(CSS) is a style sheet language ...</p>
  <a id="javascript"></a>
  <h4>JavaScript</h4>
  <p>JavaScript is a programming language for the web ...</p>
</body>
```

นอกจากการสร้างลิงก์ในลักษณะดังที่กล่าวมาทั้งหมดแล้ว ยังมีเทคนิคอื่นๆ เกี่ยวกับลิงก์ที่เราควรรู้จักเพิ่มเติม ดังต่อไปนี้

- บางครั้ง ลิงก์ที่เราสร้างขึ้นอาจยังไม่พร้อมใช้งาน ก็ให้ระบุค่าของแอตทริบิวต์ href เป็นเครื่องหมาย # ซึ่งเมื่อคลิกที่ลิงก์ จะยังอยู่ที่เพจเดิม เช่น

```
<a href="#">ประกาศรายชื่อผู้โชคดี</a>
```

- เราอาจใช้รูปภาพเป็นลิงก์แทนข้อความ โดยวางแท็ก ซ้อนในแท็ก <a> เช่น

```
<a href="http://www.developerthai.com"></a>
```

- เราสามารถใช้แท็ก <a> ในการสร้างลิงก์เพื่อเรียก JavaScript ขึ้นมาทำงานได้ โดยระบุคำสั่ง JavaScript ให้เป็นค่าของแอตทริบิวต์ href เช่น

```
<a href="javascript: alert('Hello')">Test</a>
```

การแสดงผลภาพบนเว็บเพจ

การแสดงผลภาพบนเว็บเพจนั้น เราจะกำหนดด้วยแท็ก ซึ่งเป็นประเภท Single Tag คือมีเฉพาะ Start Tag โดยมีแอตทริบิวต์ที่สำคัญคือ

inspiration starts here

```

```

สำหรับแท็ก อย่างน้อยต้องกำหนดแอตทริบิวต์ src เพื่อระบุชื่อภาพ ทั้งนี้หากภาพอยู่ในเว็บไซต์เดียวกัน เราอาจกำหนดแค่ชื่อ (รวมถึงพาธ) ของมัน เช่น

```

```

แต่หากอยู่บนเว็บไซต์อื่นๆ ต้องระบุ URL แบบเต็ม (หรือแม้จะอยู่ในเว็บไซต์เดียวกันก็ใช้วิธีนี้ได้) เช่น

```

```

การเล่นไฟล์มัลติมีเดียบนเว็บเพจ

ใน HTML5 มีแท็ก <video> สำหรับใช้เล่นไฟล์ประเภทวิดีโอโดยตรง แต่ชนิดของไฟล์ที่เว็บเบราว์เซอร์รองรับครบทุกตัวคือ mp4 เท่านั้น ซึ่งแท็ก <video> มีแอตทริบิวต์ที่สำคัญดังนี้

```
<video src="ชื่อไฟล์" width="ความกว้าง" height="ความสูง"
  preload="none | auto | metadata"
  controls    (กำหนดเมื่อต้องการแสดงปุ่มควบคุมการเล่นวิดีโอ)
  loop       (กำหนดเมื่อต้องการเล่นไฟล์ซ้ำ)
  autoplay   (กำหนดเมื่อต้องการเริ่มเล่นไฟล์โดยอัตโนมัติ)
  muted      (กำหนดเมื่อไม่ต้องการให้มีเสียง)
>
```

- แอตทริบิวต์ที่ต้องมีเสมอคือ src ส่วนแอตทริบิวต์อื่นๆ ขึ้นกับความต้องการของเราเอง
- แอตทริบิวต์ที่เป็นแบบบูลีน หากเราต้องการกำหนดให้มีแอตทริบิวต์นั้น ก็ให้ระบุชื่อแอตทริบิวต์นั้นลงไปโดยไม่ต้องกำหนดค่า เช่น <video src="video.mp4" controls loop autoplay>
- แอตทริบิวต์ **preload** กำหนดค่าได้ 3 อย่างคือ
 - **none** ไม่โหลดวิดีโอล่วงหน้า จนกว่าผู้ใช้จะคลิกปุ่มเริ่มเล่น
 - **auto** โหลดวิดีโอไว้ล่วงหน้าทันที (ค่าดีฟอลต์)
 - **metadata** โหลดเฉพาะข้อมูลเกี่ยวกับวิดีโอ เช่น ขนาดความกว้าง × สูง, เฟรมแรก หรือความยาว (เวลา) เป็นต้น
- แอตทริบิวต์ **preload** ถือเป็นทางเลือกสำหรับกรณีที่เราไม่กำหนด autoplay แต่ถ้าเรากำหนดแอตทริบิวต์ **autoplay** แล้ว ไม่ต้องกำหนดแอตทริบิวต์นี้อีก

bvn/html5/video.html

```
...
<body>
  <h3>เล่นไฟล์ video</h3>
  <video src="big-buck-bunny.mp4" width="320" height="240"
    preload="auto" controls loop
  >
  </video>
</body>
```

ส่วนกรณีของการเล่นไฟล์ประเภทเสียง ก็มีแท็กให้ใช้งานโดยตรงคือ `<audio>` ซึ่งชนิดของไฟล์ที่รองรับ ได้แก่ MP3, Wave และ Ogg โดยแอตทริบิวต์ที่สำคัญของแท็ก `<audio>` เช่น `src`, `controls`, `loop`, `autoplay`, `preload`, `muted` ก็มีรายละเอียดเช่นเดียวกับแท็ก `<video>`

bvn/html5/audio.html

```
...
<body>
  <h3>เล่นไฟล์ audio</h3>
  <audio src="beat.mp3" preload="auto" controls></audio>
</body>
```

การสร้างตาราง

ตาราง (Table) ใช้ในการจัดโครงสร้างและเนื้อหาของเพจมาให้เป็นระเบียบเรียบร้อยและสวยงาม โดยแท็กที่เกี่ยวข้องกับการสร้างตารางมีดังนี้

<code><table></code>	ใช้กำหนดขอบเขตของตาราง	<code><caption></code>	ใช้กำหนดป้ายชื่อตาราง
<code><thead></code>	ใช้กำหนดส่วนหัวตาราง	<code><tr></code>	กำหนดแถวของตาราง
<code><tbody></code>	ใช้กำหนดส่วนข้อมูลของตาราง	<code><th></code>	ใช้กำหนดเซลล์หัวตาราง
<code><tfoot></code>	ใช้กำหนดส่วนล่างของตาราง	<code><td></code>	กำหนดช่องเซลล์ของตาราง

แนวทางการสร้างตารางแบบง่ายๆ เพื่อแสดงข้อมูลบางส่วนเกี่ยวกับสินค้า โดยใช้แท็กเท่าที่จำเป็นกันไปก่อน เช่น

bvn/html5/table1.html

```
...
<body>
  <table border="1">
    <caption>รายการสินค้า</caption>
    <tr><th>Product</th><th>Price</th><td>Spec</td></tr>
    <tr><td>iPhone</td><td>40,000</td><td>6.4", 64BG, Red</td></tr>
    <tr><td>iPad</td><td>3,2000</td><td>11", 256GB, Silver</td></tr>
    <tr>
      <td>MacBook Pro</td>
      <td>100,000</td>
```

รายการสินค้า		
Product	Price	Spec
iPhone	40,000	6.4", 64BG, Red
iPad	3,2000	11", 256GB, Silver
MacBook Pro	100,000	15", 512GB, Core i7, Space Gray

bvn/html5/table1.html (ต่อ)

```

        <td>15", 512GB, Core i7, Space Gray</td>
      </tr>
    </table>
  </body>

```

ตารางนี้กำหนดขนาดเส้นขอบ (แอตทริบิวต์ border) ให้เป็น 1 เพื่อให้เห็นแนวการแบ่งแถวและคอลัมน์ที่ชัดเจนขึ้น โดยแอตทริบิวต์ border ต้องใช้กับแท็ก <table> เท่านั้น

นอกจากนี้ เรายังสามารถจัดแบ่งกลุ่มแถวของตารางด้วยแท็กดังต่อไปนี้

- แท็ก <thead> สำหรับกำหนดกลุ่มแถวที่เป็นส่วนหัวของตาราง
- แท็ก <tbody> สำหรับกำหนดกลุ่มแถวที่เป็นส่วนข้อมูลของตาราง
- แท็ก <tfoot> สำหรับกำหนดกลุ่มแถวที่เป็นส่วนล่างของตาราง

ในแต่ละกลุ่ม จะมีจำนวนกี่แถวก็ได้ และไม่จำเป็นต้องแบ่งครบทุกกลุ่ม เช่น อาจมีเพียงกลุ่ม <thead> และ/หรือ <tbody> เป็นต้น ซึ่งลักษณะการนำไปใช้งาน เช่น จากตารางในโค้ดที่ผ่านมา หากเราต้องการจัดกลุ่ม ก็อาจแก้ไขเป็นดังนี้

bvn/html5/table2.html

```

...
<body>
  <table border="1">
    <caption>รายการสินค้า</caption>
    <thead>
      <tr><th>Product</th><th>Price</th><th>Spec</th></tr>
    </thead>
    <tbody>
      <tr><td>iPhone</td><td>40,000</td><td>6.4", ...d</td></tr>
      ...
    </tbody>
  </table>
</body>

```

แม้ผลลัพธ์ที่ได้จะไม่แตกต่างจากการไม่จัดกลุ่มแถวเลย แต่ในบางกรณี เช่น การใช้ร่วมกับ Bootstrap เราจำเป็นต้องจัดกลุ่มแถวของตาราง เพื่อกำหนดรูปแบบเฉพาะในแต่ละกลุ่ม ดังที่จะกล่าวถึงในบทต่อไป

พื้นฐานที่สำคัญ ของ CSS3

2

CSS (Cascading Style Sheets) เป็นคำสั่งที่ใช้ในการจัดรูปแบบเอกสารเว็บเพจในลักษณะต่างๆ เช่น สี พื้นหลัง เส้นขอบ ขนาด โครงร่าง หรือตัวอักษร เป็นต้น แม้ว่าใน HTML นั้นจะมีทั้งแท็กและแอตทริบิวต์บางส่วนที่สามารถใช้จัดรูปแบบเอกสารอยู่แล้ว แต่ไม่ครอบคลุมการใช้งานครบทุกกรณี นอกจากนี้ก็ยังขาดความยืดหยุ่นและไม่สามารถจัดรูปแบบในลักษณะที่ซับซ้อนได้ ซึ่งในปัจจุบันเราจะเน้นการจัดรูปแบบด้วย CSS มากกว่าที่จะใช้แท็กและแอตทริบิวต์ของ HTML ดังนั้น จึงเกิดการแยกหน้าที่กันอย่างชัดเจนคือ HTML ใช้สำหรับการจัดโครงสร้างของเว็บเพจ ส่วน CSS ใช้สำหรับการจัดรูปแบบของเว็บเพจ สำหรับบทนี้เราจะได้ศึกษาการจัดรูปแบบเอกสารด้วย CSS ที่สำคัญในเบื้องต้น เพื่อเป็นพื้นฐานไปสู่การใช้ Bootstrap ที่จะกล่าวถึงในบทต่อไป

พร็อพเพอร์ตี้ของ CSS

พร็อพเพอร์ตี้ (Property) ของ CSS คือสิ่งที่ใช้กำหนดคุณลักษณะต่างๆ ของอิลิเมนต์ HTML เช่น ทำให้ข้อความเป็นสีฟ้า + พื้นหลังเป็นสีเหลือง หรือทำให้ฟอนต์มีขนาดตามที่ต้องการ เป็นต้น สำหรับหลักการเบื้องต้นเกี่ยวกับพร็อพเพอร์ตี้ที่เราควรทราบ มีดังนี้

การกำหนดค่าของพร็อพเพอร์ตี้

รูปแบบการกำหนดค่าของพร็อพเพอร์ตี้จะมี 2 ส่วนคือ ชื่อพร็อพเพอร์ตี้ และค่าของมัน โดยคั่นระหว่างชื่อและค่าด้วยเครื่องหมาย Colon (:) คือ

ชื่อพร็อพเพอร์ตี้: ค่าที่กำหนด

กรณีที่มีมากกว่า 1 พร็อพเพอร์ตี้ ให้คั่นแต่ละพร็อพเพอร์ตี้ด้วยเครื่องหมาย Semicolon (;) ในรูปแบบดังต่อไปนี้

```
พร็อพเพอร์ตี้-1: ค่าที่กำหนด; พร็อพเพอร์ตี้-2: ค่าที่กำหนด; พร็อพเพอร์ตี้-3: ค่าที่กำหนด; ...
```

แนวทางการกำหนดพร็อพเพอร์ตี้ เช่น

```
color: white; background-color: black; font-size: 12pt;
```

ถ้าเราไม่เขียนรวมไว้ในบรรทัดเดียวกัน เราอาจเขียนแยกกันไว้คนละบรรทัดกันก็ได้ เนื่องจากมีเครื่องหมาย ; คั่นระหว่างพร็อพเพอร์ตี้อยู่แล้ว เช่น

```
color: white;
background-color: black;
font-size: 12pt;
```

พร็อพเพอร์ตี้พื้นฐานที่ควรรู้จัก

ใน CSS นั้น มีพร็อพเพอร์ตี้ที่ใช้กำหนดรูปแบบเป็นจำนวนมาก ซึ่งเราจะได้ศึกษาต่อเนื่องกันไปเรื่อยๆ แต่ในหัวข้อนี้นี้ เราควรรู้จักกับพร็อพเพอร์ตี้พื้นฐานบางส่วนของ CSS ล่วงหน้าไว้ก่อน เพื่อใช้ประกอบการศึกษาร่วมกับหัวข้อในช่วงแรกๆ ของบทนี้ นั่นคือ

color	กำหนดสีข้อความ โดยในเบื้องต้นให้ระบุค่าเป็นชื่อสีในภาษาอังกฤษไปก่อน เช่น green, blue, red, yellow, black, white, gray, pink, brown, orange เป็นต้น
background-color	กำหนดสีพื้นหลัง โดยระบุค่าเป็นชื่อสีเช่นเดียวกับ color
font-size	กำหนดขนาดของฟอนต์ ซึ่งในเบื้องต้นให้กำหนดค่าเป็นตัวเลข ตามด้วยหน่วย pt หรือ px โดยตัวเลขและหน่วยให้เขียนติดกัน เช่น font-size: 10pt หรือ font-size: 12px
font-weight	กำหนดขนาดน้ำหนักของฟอนต์ มักใช้เมื่อต้องการทำให้ข้อความเป็นตัวหนา โดยระบุค่าเป็น bold เช่น font-weight: bold
font-style	กำหนดรูปแบบฟอนต์ โดยทั่วไปมักกำหนดเมื่อต้องการทำให้ข้อความเป็นตัวเอียง โดยระบุค่า italic เช่น font-style: italic
text-decoration	สำหรับตกแต่งข้อความ มักใช้เมื่อต้องการขีดเส้นใต้ โดยระบุค่าเป็น underline เช่น text-decoration: underline

ขอบเขตของการกำหนดสไตล์แบบต่าง ๆ

การกำหนดสไตล์แบ่งออกเป็น 3 แบบ คือ

- สไตล์แบบ Inline จะมีผลเฉพาะกับอิลิเมนต์ที่กำหนดสไตล์ให้กับมัน
- สไตล์แบบ Internal จะมีผลกับทุกอิลิเมนต์ภายในเพจนั้น ที่ตรงกับข้อกำหนด
- สไตล์แบบ External นำเข้าสไตส์จากไฟล์ภายนอก และจะมีผลกับทุกอิลิเมนต์ภายในเพจนั้น ที่ตรงกับข้อกำหนด

สำหรับรายละเอียดการกำหนดสไตล์แต่ละแบบมีดังต่อไปนี้

การกำหนดสไตล์แบบ Inline

สไตส์แบบ Inline คือการกำหนดรูปแบบเอาไว้ที่แท็กซึ่งเราต้องการจัดรูปแบบ โดยระบุพร็อพเพอร์ตี้และค่าของมันให้กับแอตทริบิวต์ style ซึ่งรูปแบบคือ

```
<แท็ก style="พร็อพเพอร์ตี้-1: ค่าที่กำหนด; พร็อพเพอร์ตี้-2: ค่าที่กำหนด; ..." >
```

แนวทางการกำหนดสไตล์แบบ Inline เช่น



การกำหนดสไตล์แบบ Internal

สไตส์แบบ Internal จะกำหนดเอาไว้ที่ส่วนหัวของเอกสาร เพื่อให้มีผลกับหลายๆ อิลิเมนต์ โดยไม่ต้องเขียนซ้ำซ้อนกันอีก ดังรูปแบบต่อไปนี้

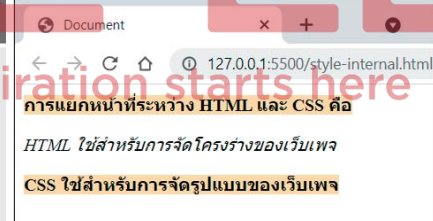
```
<style>
  ซีเล็คเตอร์ {
    พร็อพเพอร์ตี้-1: ค่าที่กำหนด;
    พร็อพเพอร์ตี้-2: ค่าที่กำหนด;
    ...
  }
</style>
```

สิ่งที่ควรรู้เพิ่มเติมเกี่ยวกับสไตลแบบ Internal คือ

- สไตลแบบ Internal ต้องกำหนดด้วยแท็ก <style> และให้อยู่ระหว่างแท็ก <head>...</head>
- ซีเล็คเตอร์ (Selector) เป็นตัวที่ใช้กำหนดว่า จะให้สไตลนั้นนี้มีผลกับอิลิเมนต์ใด โดยมีซีเล็คเตอร์หลายรูปแบบ เช่น Type, ID, Class เป็นต้น ซึ่งรายละเอียดเกี่ยวกับซีเล็คเตอร์แต่ละแบบ จะกล่าวถึงในหัวข้อต่อไป
- พร็อพเพอร์ตี้และค่าของมัน ต้องกำหนดไว้ในวงเล็บ { } เสมอ
- เราอาจเขียนพร็อพเพอร์ตี้รวมไว้ในบรรทัดเดียวกันดังโค้ดที่แล้ว หรือแยกบรรทัดกันก็ได้

bvn/css3/style-internal.html

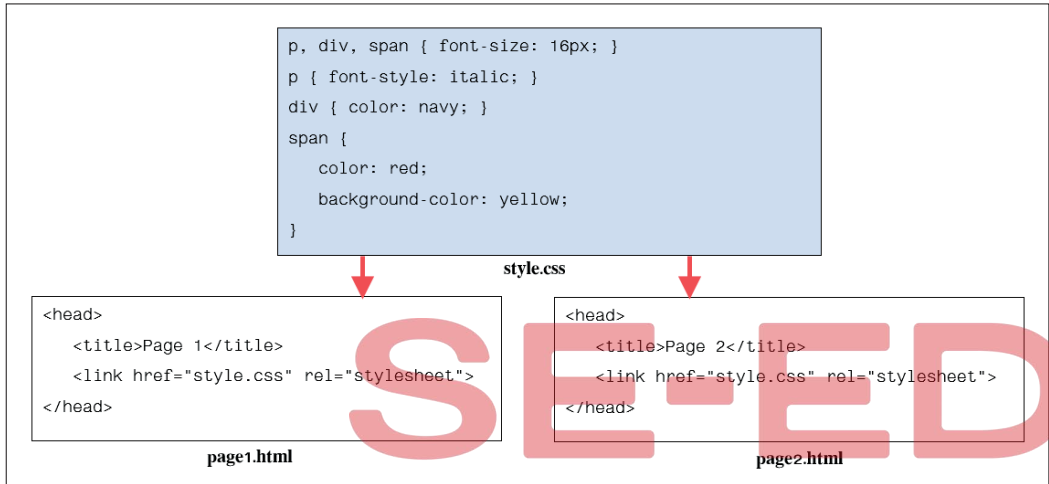
```
<head>
  ...
  <style>
    div { font-size: 12pt; }
    p { font-style: italic; }
    span {
      background-color: wheat;
      font-weight: bold;
    }
  </style>
</head>
<body>
  <div>
    <span>การแยกหน้าที่ระหว่าง HTML และ CSS คือ</span>
    <p>HTML ใช้สำหรับการจัดโครงสร้างของเว็บเพจ</p>
    <span>CSS ใช้สำหรับการจัดรูปแบบของเว็บเพจ</span>
  </div>
</body>
```



การกำหนดสไตล์แบบ External

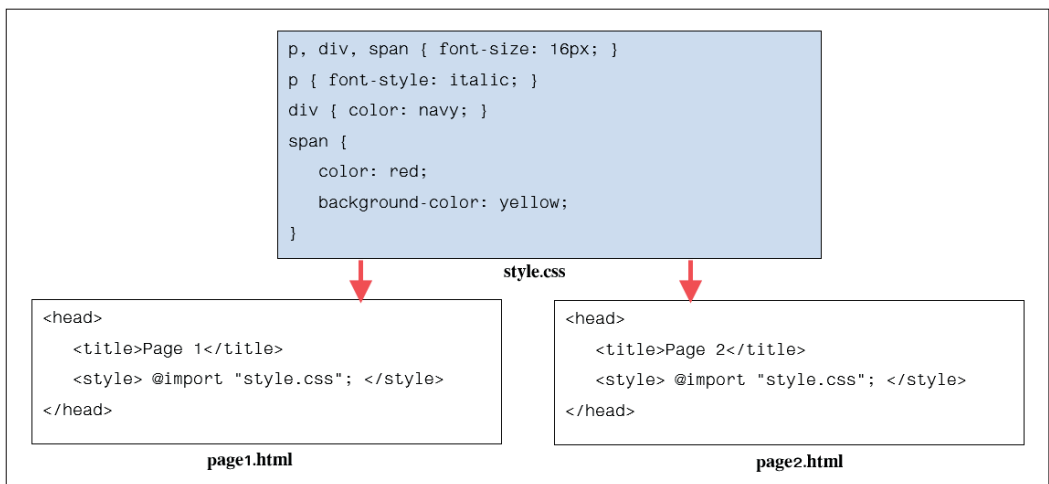
สไตล์แบบ External มักใช้กับกรณีที่เราจำเป็นต้องใช้สไตล์แบบเดียวกันในหลายๆ เพจ โดยแยกเฉพาะส่วนสไตล์ที่ซ้ำกันไปสร้างไว้ที่ไฟล์ภายนอก แล้วเมื่อต้องการใช้ที่เพจใด ก็ทำการเชื่อมโยงกับไฟล์นั้น ซึ่งมีให้เลือก 2 วิธีคือ

- **วิธีที่ 1** เชื่อมโยงโดยวางแท็ก <link> ไว้ที่ส่วนหัวของเพจซึ่งต้องการใช้สไตล์ ดังภาพ



inspiration starts here

- **วิธีที่ 2** เชื่อมโยงด้วยคีย์เวิร์ด @import โดยวางไว้ในแท็ก <style> ของเพจที่ต้องการใช้สไตล์ ดังภาพ



พัฒนา Web App แบบ Responsive

ด้วย Bootstrap5

ร่วมกับ Vue.js และ Node.js

หนังสือเล่มนี้ ได้รวบรวมสิ่งที่จำเป็นต้องรู้ทั้งหมดสำหรับการพัฒนาเว็บแอปพลิเคชันเอาไว้อย่างครบถ้วน นับตั้งแต่การจัดโครงสร้างของเพจด้วย HTML5 การปรับแต่งรูปแบบการแสดงผลให้สวยงามน่าใช้งาน พร้อมทั้งรองรับการแสดงผลบนหน้าจอที่มีขนาดแตกต่างกันด้วย CSS3 และ Bootstrap5 รวมถึงการควบคุมเนื้อหาให้ตรงตามเงื่อนไขด้วย Vue.js ตลอดจนการรับส่งข้อมูลและประมวลผลทางฝั่งเซิร์ฟเวอร์โดยใช้ Node.js พร้อมทั้งไลบรารีที่เกี่ยวข้องอีกหลายอย่าง ซึ่งผู้อ่านสามารถนำไปประยุกต์ใช้ร่วมกันเพื่อพัฒนาเว็บแอปพลิเคชันที่สามารถใช้งานได้จริงตามที่ต้องการ

เนื้อหาในหนังสือประกอบด้วย :

- บทที่ 1 VS Code และพื้นฐาน HTML5
- บทที่ 2 พื้นฐานที่สำคัญของ CSS3
- บทที่ 3 BS5: การติดตั้งและใช้คลาสพื้นฐาน
- บทที่ 4 BS5: การใช้คอมโพเนนต์ (1)
- บทที่ 5 BS5: การใช้คอมโพเนนต์ (2)
- บทที่ 6 BS5: การจัดรูปแบบของฟอร์ม
- บทที่ 7 BS5: การแสดงผลแบบ Flex
- บทที่ 8 BS5: จัดโครงสร้างแบบ Responsive
- บทที่ 9 ทบทวน JavaScript และ Node.js

- บทที่ 10 Vue: การใช้งานเบื้องต้นและอีเวนต์
- บทที่ 11 Vue: การใช้ Directive และ Binding
- บทที่ 12 Vue: การใช้ V-Model ร่วมกับฟอร์ม
- บทที่ 13 Vue: การสร้างและใช้งานคอมโพเนนต์
- บทที่ 14 Vue: การสร้างแอปในแบบ Vue CLI
- บทที่ 15 Vue: การกำหนดเส้นทางด้วย Router
- บทที่ 16 Node: การรับส่งข้อมูลกับเซิร์ฟเวอร์
- บทที่ 17 Node: ฟอร์ม การอัปโหลด และ Session
- บทที่ 18 Node: การใช้งานร่วมกับฐานข้อมูล

SE-ED
inspiration starts here



www.se-ed.com



sbc.fans



SE-ED Publisher

พร้อมจำหน่ายในรูปแบบ

- | | |
|--|---|
| ☒ e-book (PDF) | ☐ audiobooks |
| ☐ e-book (EPUB) | ☐ audio CD / MP3 |
| ☒ ปกอ่อน | ☐ LARGE PRINT (ตัวอักษรขนาดใหญ่) |

ISBN 978-616-08-4383-1



9 786160 843831

299 บาท

คอมพิวเตอร์-การเขียนโปรแกรม