FutureSkill

HTML CHEATSHEET

เริ่มต้นสร้างและแสดงเนื้อหาของเว็บไซต์ด้วย HTML

โครงสร้างพื้นฐาน

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <meta name="viewport"
content="width=device-width, initial-
scale=1.0" />
    <title>Document</title>
  </head>
  <body>
    <!-- … -->
  </body>
</html>
```

<element attribute="value">...</element>
แท็ก HTML เรียกอีกอย่างได้ว่า element ภายในสามารถมี attribute และ value ได้ เช่น ... จะมี a เป็น element, href เป็น attribute, https://example.com เป็น value ของ href ว่าจะให้ลิงก์ไปที่ไหน

<!-- คอมเมนต์ -->
ผู้ใช้จะไม่เห็นบนหน้าเว็บ

<title>...</title>
ใส่ชื่อให้เว็บไซต์

สำหรับเนื้อหา

ลิงก์
ลิงก์ไปหน้าเว็บที่จะไป

- href="#elementId"
- ลิงก์ไปที่ element ที่มี id="elementId"

<h1>หัวข้อ</h1>
หัวข้อสามารถมีได้ตั้งแต่ h1 – h6 แต่โดยทั่วไป h1 – h3 ก็เพียงพอแล้ว

ตัวหนา
เน้นข้อความให้เป็นตัวหนา

ตัวเอียง
เน้นข้อความ (emphasis) โดยปกติบราวเซอร์จะเขียนออกมาเป็นตัวเอียง

ข้อความยาว ๆ เป็นย่อหน้า…
จัดกลุ่มข้อความยาว ๆ เป็นแต่ละย่อหน้า (paragraph) โดยภายในแท็ก … ควรมีแค่ 1 ย่อหน้า

 ลิสต์เรียงลำดับ,

 ลิสต์ทั่วไป

 (ordered list) จะสร้างลิสต์ 1, 2, 3, …
- start="10" ให้เริ่มที่ 10 แทนที่จะเริ่มที่ 1

 (unordered list) จะสร้างลิสต์ที่เป็น bullet point (จุดไข่ปลา) นำหน้า
โดยแต่ละหัวข้อต้องอยู่ในแท็ก … เช่น

```
<ol>
  <li>หัวข้อ 1</li>
  <li>หัวข้อ 2</li>
</ol>
```


สำหรับใส่ภาพ

สำหรับจัดกลุ่ม

…
จัดกลุ่ม elements ที่เกี่ยวข้องเข้าด้วยกัน แสดงผลแบบ block (ไม่สามารถอยู่บรรทัดเดียวกับคนอื่นได้)

…
จัดกลุ่ม elements เข้าด้วยกัน แสดงผลแบบ inline (สามารถอยู่ร่วมบรรทัดกับ inline อื่นได้)

<header>…</header>
(semantic) จัดกลุ่ม elements ที่เกี่ยวกับส่วนหัวของหน้าเว็บไซต์ เช่น โลโก้, แถบลิงก์ด้านบน, กล่องค้นหา

<main>…</main>
(semantic) จัดกลุ่มเนื้อหาหลักของเว็บไซต์ ใน 1 หน้าสามารถมี <main> ได้เพียง 1 ครั้ง

<section>…</section>
(semantic) จัดกลุ่มให้ elements ที่เกี่ยวข้องกลับส่วนนั้น ๆ อยู่ด้วยกัน เช่น ส่วนคอมเมนต์, ส่วนรีวิว

<footer>…</footer>
(semantic) จัดกลุ่มสำหรับเนื้อหาส่วนท้ายของเว็บไซต์ เช่น ลิขสิทธิ์, ที่อยู่, ลิงก์โซเชียล

ฟอร์ม

<form>…</form>
ครอบฟอร์ม และ input ต่าง ๆ

```
<label>
  …
  <input … />
</label>
```

สำหรับบอกว่า input นั้นคืออะไร เช่น <label>รหัสผ่าน <input … /></label>
<input type="…" />

- required

บังคับว่า input นั้นต้องถูกกรอก (เป็น attribute พิเศษที่ไม่ต้องมี ="value" ต่อท้าย)

- placeholder="…"

สำหรับเป็นตัวช่วยให้ผู้ใช้เข้าใจ input มากขึ้น เช่น placeholder="example@email.com"

- name="…"

ชื่อของช่อง input

- type="text"

รับข้อความทั่วไป

- type="password"

สำหรับรหัสผ่าน

- type="email"

สำหรับอีเมล

- type="checkbox"

กล่องเลือกถูก-ผิด สามารถเลือกมากกว่า 1 กล่องที่มี name เดียวกันได้ (เช่น ช่องทางรับข่าวสาร)

- type="radio"

สำหรับเลือกตัวเลือกเดียวที่มี name เดียวกัน (เช่น เพศ, อายุ)

- value="…"

ค่าของ input นั้น (เช่น <input type="checkbox" value="male" />)

```
<select name="…">
  <option value="…">…</option>
</select>
```

สร้าง drop-down menu สำหรับตัวเลือกเยอะ ๆ

ตาราง

<table>…</table>
สร้างตาราง

<thead>…</thead>
<tbody>…</tbody>
<tfoot>…</tfoot>
(semantic) บอกส่วนหัว ส่วนหลัก และส่วนท้ายของตาราง

<tr>…</tr>
สร้างแถว (แนวนอน) ของตาราง

<th>…</th> (ช่องหัวข้อ)
<td>…</td> (ช่องเนื้อหา)
สร้างคอลัมน์ (แนวตั้ง) ของตาราง (ต้องอยู่ใน <tr>…</tr>)

rowspan="2"
ขยายรวมกับแถวอื่นเพิ่มเป็น 2 แถว

colspan="2"
ขยายรวมกับคอลัมน์อื่นเพิ่มเป็น 2 แถว

<table>
<tr><td><tr></td><td><th>...</th></td><td><th>...</th></td><td><th>...</th></td><td></tr></td></tr>
<tr><td><tr></td><td colspan="2"><td colspan="2">...</td></td><td rowspan="2"><td rowspan="2">...</td></td><td></tr></td></tr>
<tr><td><tr></td><td><td>...</td></td><td><td>...</td></td><td></tr></td></tr>
</table>

FutureSkill

CSS CHEATSHEET

ปรับแต่งและจัดวางสไตล์ของเว็บไซต์ด้วย CSS

การ import CSS ใน HTML

```
<!DOCTYPE html>
<html lang="en">
        <head>
                <link rel="stylesheet" href="[...].css" />
        </head>
        <body>
                <!-- ... -->
        </body>
</html>
```

/* comment */

CSS Selectors

CSS Selector	Element	
.classSelector	<... class="classSelector">	ทุก element ที่มี class เป็น classSelector
#idSelector	<... id="idSelector">	Element ที่มี id เป็น idSelector
*	ทุก element	ทุก element
div		ทุก
div, p	,	ทุก และ
div.classSelector		ที่มี class เป็น classSelector
div p	... ...	ทุก ที่อยู่ใน ...
div > p		ทุก ที่อยู่ใน ...
div + p		ทุก ที่อยู่หลัง
[attribute=value]	<... attribute="value">	ทุก element ที่มี attribute="value"

Properties พื้นฐาน

background

- background-color: #123456;
  background-color: red;
  background-color: rgba(0, 127, 255, 0.5);
  ใส่สีพื้นหลัง
- background-image: url('https://...');
  ใส่ภาพพื้นหลัง
- background-size: cover;
  ให้ภาพพื้นหลังเต็มกล่อง (อาจจะไม่เห็นครบทั้งภาพ)
- background-size: cover;
  ให้เห็นภาพพื้นหลังครบทั้งภาพ (ภาพอาจซ้ำ และมีที่ว่างเหลือไม่เต็มกล่อง)
- background-size: 50%;
  ให้ภาพพื้นหลังมีขนาด 50%

color

ใส่สีให้ตัวอักษร

font

- font-family: 'Open Sans', Helvetica, Arial, sans-serif;

กำหนดชื่อฟอนต์

- font-size: 16px;

กำหนดขนาดฟอนต์

- font-weight: 400;

กำหนดน้ำหนัก (ความหนา) ฟอนต์ (400: ปกติ, 700: ตัวหนา)

width, height

กำหนดความกว้าง, ความสูง สามารถมีหน่วยเป็น px (pixel), % หรืออื่น ๆ เช่น em, rem ได้

border

- border-size: 1px;

ความหนาของเส้นขอบ

- border-color: #123456;
- border-color: red;
- border-color: rgba(0, 127, 255, 0.5);

สีของเส้นขอบ

- border-style: solid/dashed/...;

สไตล์ของเส้นขอบ (เส้นทึบ, เส้นประ ฯลฯ)

- border-radius: 8px;

ความโค้งมนของขอบ

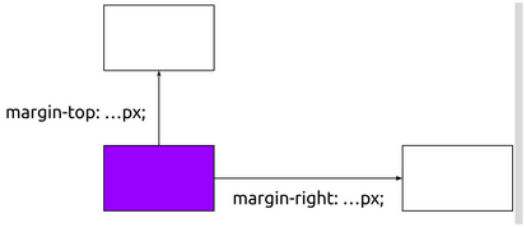
div > p ต่างกับ div p อย่างไร?

```
<div>
  <span>
    <p id="p1"> P1 </p> <!-- div p -->
  </span>
  <p id="p2"> P2 </p> <!-- div p และ div > p -->
</div>
```

จากโค้ดด้านบน div p จะเลือกได้ทั้ง P1, P2 (ทุก p ที่อยู่ใน div)
แต่ div > p จะเลือกได้เฉพาะ P2 (ทุก p ที่อยู่ใน div เท่านั้น ไม่สามารถอยู่ใน <> </> อื่นได้)

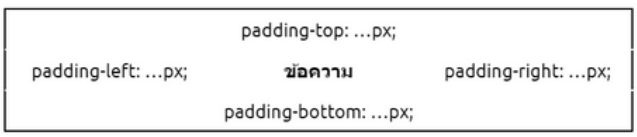
margin

กำหนดระยะห่างจาก element รอบข้าง สามารถระบุทิศทางได้ เช่น margin-top, -right, -bottom, -left



padding

กำหนดระยะห่างภายในกล่อง



display

- display: block;

ให้กล่องเป็น block อยู่ร่วมกับ element อื่นในบรรทัดเดียวกันไม่ได้

- display: inline;

ให้กล่องเป็น inline สามารถอยู่ร่วมกับ element อื่นในบรรทัดเดียวกันได้

FutureSkill

JAVASCRIPT CHEATSHEET

เพิ่มลูกเล่นให้กับเว็บไซต์ด้วย JavaScript

พื้นฐาน

การ import JavaScript ใน HTML

```
<!DOCTYPE html>
<html lang="en">
        <head>
                <!-- ... -->
        </head>
        <body>
                <!-- ... -->
                <script type="text/javascript" src="[...].js"></script>
        </body>
</html>
```

```
// comment แบบบรรทัดเดียว
/*
   คอมเมนต์แบบหลายบรรทัด
*/
```

Hello, world

- console.log('Hello, world!');
  พิมพ์ลงใน console (เปิด dev tools ด้วย คลิกขวา > inspect)
- alert('Hello, world!');
  เด้ง pop up พร้อมข้อความ
- document.querySelector('body')
  .innerText = 'Hello, world!';
  ใส่ Hello, world! ลงไปในหน้าเว็บ

ตัวแปร

- const foo = 1234;

ประกาศตัวแปร ไม่สามารถประกาศค่าทับลงไปได้ เช่น

```
const quz = 'hello';
quz = 'world'; // ไม่ได้!
```

- let bar = 6789;

ประกาศตัวแปรที่สามารถเขียนค่าทับลงไปได้ เช่น bar = 101112; จะไม่มีปัญหา เช่น

```
let qux = 'hello';
qux = 'world'; // ได้ (qux มีค่าเป็น 'world')
```

DOM

- document.querySelector('#id1');

หา element แรกที่มี id="id1"

- document.querySelectorAll('.class1');

หาทุก element ที่มี class="class1"

- document.createElement('div');

สร้าง element div ขึ้นมาใหม่

- element1.append(element2, element3, ...);

ใส่ element2, element3, ... ลงไปใน element1 ผลลัพธ์:

```
<element1>
   <element2></element2>
   <element3></element3>
</element1>
```

- element.className = 'class1 class2';

ให้ element มี class="class1 class2"

- element.classList

ลิสต์ class ของ element เช่น <element class="class1 class2"> จะได้เป็น ['class1', 'class2']

- element.classList.add('classA')

เพิ่ม classA ลงไปใน element

- element.classList.remove('classB')

ลบ classB ออกจาก element

- const sayHello = () => { ... }

element.addEventListener('click', sayHello);

เพิ่ม eventListener ลงไปว่า ถ้ามีเหตุการ click บน element นี้ ให้ลองทำตามฟังก์ชัน sayHello ที่ส่งเข้าไป

ดู events ได้จาก https://developer.mozilla.org/en-US/docs/Web/API/Element#events

Data types (ประเภทข้อมูล)

string ('hello!')

ข้อมูลประเภทข้อความ ตัวอักษร เช่น 'hello', '1234' ใช้เครื่องหมาย '...' ในการบอกว่าเป็น string

การต่อ string เข้ากับตัวแปรอื่น

- **'hello' + name**
  ใช้เครื่องหมาย + ในการต่อ string เข้ากับตัวแปรอื่น
- **`hello ${name}`**

สามารถใช้ template literal ในการใส่ตัวแปรลงไปใน string ได้

number (42)

ข้อมูลประเภทตัวเลข เช่น **3.141592, 206265, -273.15**

boolean (true/false)

จริง หรือ เท็จ เหมาะสำหรับใช้ในการเขียน if-else เช่น const isAdmin = true;

array ([1, 2, 3])

การเก็บข้อมูลหลาย ๆ ตัวเข้าด้วยกันเป็นเหมือนขบวนรถไฟ โดยแต่ละตู้มีค่าเป็นอะไรก็ได้

ค่า	322	42	206265	-1234	9
Index	0	1	2	3	4

object ({ user: 'thammarith' })

เป็นการรวมข้อมูลไว้ด้วยกัน เช่น กล่องมีสีแดง ความสูง 10cm ยาว 20cm
สามารถเขียนได้เป็น

```
const box = {
   colour: 'red',
   dimension: {
      height: 10,
      width: 20,
      unit: 'cm',
   }
};
```

โดยชื่อของค่าเรียกว่า property/key
(เช่น colour, dimension, height) ส่วนตัวค่าเรียก value
(เช่น 'red', 10, 20)
สามารถอ่านค่าได้โดยใช้ ตัวแปร ['property'] เช่น box['colour']

Function

ฟังก์ชันมีไว้เพื่อลดการเขียนโค้ดซ้ำ ๆ ฟังก์ชันสามารถรับ arguments หรือไม่รับก็ได้ จะ return หรือไม่ return ค่าก็ได้

```
function double(number) {
  return number * 2;
}
```

ฟังก์ชัน double รับ argument ที่ชื่อว่า number แล้ว return ค่าที่ x2 กลับไป

```
const double = (number) => {
   return number * 2;
}
```

ฟังก์ชันเดียวกัน แต่เขียนเป็น arrow function

Condition (if, else)

การเปรียบเทียบเงื่อนไข

- 1 === 2 // false

=== เปรียบเทียบทั้งค่าว่าเท่ากันหรือไม่ และประเภทข้อมูลว่าแบบเดียวกันหรือไม่
เช่น 1 === 1 → true แต่ 1 === '1' → false

- >, >=, <, <=

สำหรับเทียบว่าฝั่งซ้ายมากกว่า, มากกว่าหรือเท่ากับ, น้อยกว่า, น้อยหว่าหรือเท่ากับฝั่งขวาหรือไม่

- && (และ)

สำหรับต่อเงื่อนไขเข้าด้วยกัน ต้องเป็นจริงทั้งหมด
เช่น 1 < 2 && 2 < 3 → true แต่ 1 < 2 && 100 < 10 → false

- || (หรือ)

สำหรับต่อเงื่อนไขเข้าด้วยกัน ตัวใดตัวหนึ่งเป็นจริงก็จะถือว่าจริง
เช่น 1 < 2 || 2 < 3 → true, 1 < 2 || 100 < 10 → true แต่ 1 < 0 || 100 < 10 → false

if (เงื่อนไข) { ... }

หากเป็นไปตามเงื่อนไข (เป็น true) ให้ทำใน { ... } เช่น

```
if (1 < 2) {
   console.log('น้อยกว่า');
}

// ผลลัพธ์: 'น้อยกว่า'
```

if (เงื่อนไข) { ... } else { ... }

เช็คเงื่อนไข ถ้าเป็นจริง ให้ทำใน { ... } ของ if ถ้าไม่เป็นจริง ให้ทำใน { ... } ของ else เช่น

```
if (1 === 2) {
   console.log('เท่ากัน');
} else {
   console.log('ไม่เท่ากัน');
}

// ผลลัพธ์: 'ไม่เท่ากัน'
```

```
if (เงื่อนไข 1) { ... }
else if (เงื่อนไข 2) { ... }
else { ... }
```

เช็คเงื่อนไข 1 ถ้าเป็นจริงทำใน { ... } ของเงื่อนไขแรก ถ้าไม่จริงให้เช็คเงื่อนไข 2 ถ้ายังไม่จริงให้ทำใน { ... } ของ else เช่น

```
const grade = 70;
if (grade >= 80) {
   console.log('A');
} else if (grade >= 70) {
   console.log('B');
} ... {
   ...
} else {
    console.log('F');
}

// ผลลัพธ์: 'B'
```