บทที่ 2 HTML5, CSS3

1. ภาพรวมของเทคโนโลยีเว็บสมัยใหม่

เทคโนโลยีเว็บในปัจจุบันได้พัฒนาไปอย่างต่อเนื่องจากยุค HTML4 สู่ HTML5 ซึ่งเป็นรากฐานสำคัญของเว็บแอปพลิเคชันยุคใหม่ที่รองรับการทำงานแบบ Cross-Platform และการใช้งานบนอุปกรณ์พกพาได้อย่างมีประสิทธิภาพ นอกจากนี้แนวคิดการออกแบบเว็บแบบ Responsive Design และ Mobile-First ยังกลายเป็นมาตรฐานในการออกแบบเว็บไซต์ที่ตอบสนองต่อความหลากหลายของอุปกรณ์และขนาดหน้าจอ

1.1 พัฒนาการจาก HTML4 สู่ HTML5

มาตรฐาน HTML 4.01 ได้รับการเผยแพร่โดย World Wide Web Consortium (W3C) ในปี 2000 เพื่อกำหนดโครงสร้างพื้นฐานของเอกสารเว็บเพจ แต่ยังคงมีข้อจำกัดหลายด้าน เช่น ไม่รองรับมัลติมีเดียโดยตรงและต้องพึ่งพา plug-in ภายนอก เช่น Adobe Flash (W3C, 2014)

ต่อมา กลุ่มนักพัฒนา WHATWG ได้เสนอให้ปรับปรุงมาตรฐาน HTML ให้สามารถรองรับแอปพลิเคชันบนเว็บได้อย่างเต็มรูปแบบ เกิดเป็น HTML5 ซึ่งเปิดตัวอย่างเป็นทางการในช่วงปี 2014 โดยมีจุดมุ่งหมายเพื่อเพิ่มประสิทธิภาพและลดข้อจำกัดของ HTML4 (Wikipedia Contributors, 2025) ด้วยเหตุนี้ HTML5 จึงกลายเป็นรากฐานสำคัญของ “เว็บสมัยใหม่” ที่รองรับการทำงานแบบ cross-platform และตอบสนองความต้องการของอุปกรณ์พกพาได้อย่างมีประสิทธิภาพ

HTML5 มีคุณสมบัติสำคัญหลายประการ ได้แก่

- โครงสร้างเชิง semantic (เช่น <header>, <nav>, <section>, <article>, <footer>) ซึ่งช่วยให้เบราว์เซอร์และเครื่องมือค้นหาทำความเข้าใจเนื้อหาได้ดีขึ้น
- การรองรับมัลติมีเดียโดยตรง ผ่านแท็ก <video> และ <audio> โดยไม่ต้องพึ่งพา plug-in
- API ใหม่ เช่น Web Storage, Canvas, Web Workers ซึ่งเปิดโอกาสให้สร้างเว็บแอปพลิเคชันที่มีความสามารถเทียบเท่าแอปพลิเคชันบนระบบปฏิบัติการ
- การเข้ากันได้ย้อนหลัง (backward compatibility) และการลดการพึ่งพา SGML

1.2 แนวคิด Responsive Design และ Mobile-First

Responsive Web Design

แนวคิด Responsive Web Design หรือ RWD ถูกเสนอโดย Ethan Marcotte ในปี 2010 โดยอธิบายว่า “เว็บไซต์ควรปรับตัวและตอบสนองต่อบริบทของผู้ใช้” (Marcotte, 2010) แนวทางนี้ใช้เทคนิค CSS เช่น fluid grid และ media queries เพื่อให้โครงสร้างและองค์ประกอบของหน้าเว็บปรับตามขนาดหน้าจอของอุปกรณ์ การใช้ media query ดังกล่าวเป็นหัวใจหลักของการออกแบบเว็บยุคใหม่ เนื่องจากช่วยลดความจำเป็นในการสร้างหลายเวอร์ชันของเว็บไซต์ (Mozilla Developer Network, n.d.)

ตัวอย่าง Responsive CSS

```
<style>
  body { font-family: "Sarabun", sans-serif; margin: 0; text-align: center; }
  .box { background-color: lightblue; padding: 40px; }

  @media (max-width: 600px) {
    .box { background-color: lightgreen; padding: 20px; }
  }
</style>
```

Mobile-First Design

แนวคิด Mobile-First Design ถือกำเนิดขึ้นเพื่อตอบสนองต่อพฤติกรรมผู้ใช้ที่เข้าถึงเว็บไซต์ผ่านสมาร์ตโฟนเป็นหลัก โดยเสนอให้เริ่มออกแบบจากหน้าจอขนาดเล็ก ก่อนขยายไปยังหน้าจอใหญ่ขึ้น (BrowserStack Team, 2024) แนวทางนี้ช่วยให้เนื้อหาสำคัญถูกนำเสนออย่างชัดเจน และเพิ่มประสิทธิภาพการโหลดหน้าเว็บ หลักการนี้สอดคล้องกับแนวคิด Progressive Enhancement ที่เริ่มจากฟังก์ชันพื้นฐานก่อน แล้วค่อยเพิ่มฟีเจอร์เมื่ออุปกรณ์รองรับ (UXPin Editorial Team, 2023)

```
<style>
  body { background-color: #f5f5f5; color: #333; text-align: center; }
  nav { background-color: #2196f3; color: white; padding: 10px; }

  @media (min-width: 768px) {
    nav { display: flex; justify-content: space-around; padding: 20px; }
  }
</style>
```

ความสัมพันธ์ระหว่าง Responsive และ Mobile-First

แม้ทั้งสองแนวคิดจะมีเป้าหมายเดียวกัน คือการทำให้เว็บไซต์เข้าถึงได้ทุกอุปกรณ์ แต่แตกต่างกันในลำดับการออกแบบ Responsive Design เน้นการปรับตัวขององค์ประกอบตามหน้าจอ ส่วน Mobile-First เน้นเริ่มออกแบบจากมือถือ แล้วขยายไปยังเดสก์ท็อป (Unified Infotech Design Team, 2024) เว็บไซต์สมัยใหม่จึงมักผสานทั้งสองแนวคิดเข้าด้วยกันเพื่อให้มีความยืดหยุ่นและประสิทธิภาพสูงสุด

2. โครงสร้างพื้นฐานของ HTML5

HTML5 คือวิวัฒนาการสำคัญของเทคโนโลยีเว็บที่ทำให้เกิดการเปลี่ยนแปลงจากเว็บเชิงเอกสาร (document-centric) ไปสู่เว็บเชิงแอปพลิเคชัน (application-centric) พร้อมทั้งสนับสนุนการออกแบบแบบ Responsive และ Mobile-First ที่มุ่งเน้นประสบการณ์ผู้ใช้บนอุปกรณ์พกพา การเข้าใจพัฒนาการดังกล่าวจึงเป็นพื้นฐานสำคัญของการเรียนการสอนในรายวิชา Mobile Web Application Development

2.1 โครงสร้างเอกสาร (Document Structure)

เอกสาร HTML5 เริ่มต้นด้วยคำประกาศ <!DOCTYPE html> ที่มีรูปแบบเรียบง่ายกว่ามาตรฐาน HTML4 (ซึ่งต้องประกาศ DTD แบบยาว) การเขียนโครงสร้างจึงกระชับ และช่วยให้เบราว์เซอร์ทุกชนิดตีความได้ตรงกัน (W3C, 2014)

ตัวอย่างโครงสร้างเอกสาร HTML5

```
<!DOCTYPE html>
<html lang="th">
  <head>
    <meta charset="UTF-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <title>โครงสร้างพื้นฐานของ HTML5</title>
  </head>
  <body>
    <header>ส่วนหัวของหน้าเว็บ</header>
    <main>
      <article>
        <h1>ยินดีต้อนรับสู่เว็บไซต์</h1>
        <p>ตัวอย่างเนื้อหาในโครงสร้าง HTML5</p>
      </article>
    </main>
    <footer>ลิขสิทธิ์ © 2025 โดย My Website</footer>
  </body>
</html>
```

คำอธิบาย:

องค์ประกอบสำคัญประกอบด้วย

- <head> เก็บข้อมูลเมตา (meta information) เช่น charset และ viewport
- <body> คือส่วนเนื้อหาหลักของหน้าเว็บ
- <main> แสดงเนื้อหาหลักของเอกสารที่แตกต่างจากส่วนหัวและส่วนท้าย
- <header> และ <footer> เป็นองค์ประกอบเชิง semantic สำหรับโครงสร้าง

โครงสร้างนี้สื่อความหมายได้ดีกว่าเดิมและสนับสนุนการเข้าถึงของผู้พิการ (accessibility) และ SEO ได้ดียิ่งขึ้น (Mozilla Developer Network, n.d.)

2.2 Semantic Tags (header, nav, article, section, footer)

องค์ประกอบ (semantic tags) ใน HTML5 มีบทบาทสำคัญในการให้ “ความหมาย” แก่ส่วนต่าง ๆ ของเอกสาร เพิ่มความชัดเจนของโครงสร้าง ช่วยให้เบราว์เซอร์และเครื่องมือค้นหาทำความเข้าใจเนื้อหาได้ถูกต้อง (Ethan Marcotte, 2010)

ตัวอย่างการใช้ Semantic Tags

```
<header>
  <h1>ข่าวเทคโนโลยี</h1>
</header>
<nav>
  <a href="#home">หน้าแรก</a>
  <a href="#tech">เทคโนโลยี</a>
  <a href="#contact">ติดต่อ</a>
</nav>
<section id="tech">
  <article>
    <h2>บทความ: HTML5 และอนาคตของเว็บ</h2>
    <p>HTML5 ช่วยให้เว็บทำงานได้คล้ายแอปพลิเคชันมากขึ้น...</p>
  </article>
</section>
<footer>
  <p>สงวนลิขสิทธิ์ © 2025 TechNews</p>
</footer>
```

ความหมายของแท็กหลัก

แท็ก **หน้าที่**

- <header> ส่วนหัวของหน้าเว็บหรือบทความ

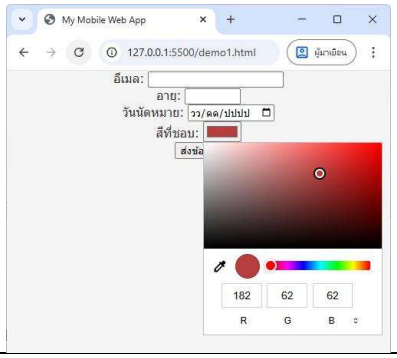
- <nav> เมนูหรือลิงก์นำทาง
- <section> กลุ่มเนื้อหาหลักภายในเอกสาร
- <article> เนื้อหาที่เป็นอิสระ เช่น บทความหรือโพสต์
- <footer> ส่วนท้ายของเอกสาร เช่น ข้อมูลลิขสิทธิ์หรือช่องทางติดต่อ

2.3 ฟอร์ม (Forms) และ Input Types ใหม่ใน HTML5

HTML5 ได้ขยายความสามารถของ ฟอร์ม โดยเพิ่มชนิดข้อมูล (input types) และ attributes ใหม่ เพื่อให้การกรอกข้อมูลมีประสิทธิภาพมากขึ้นและลดภาระของ JavaScript

ตัวอย่างฟอร์ม HTML5

```
<form>
  <label for="email">อีเมล:</label>
  <input type="email" id="email" name="email" required /><br>
  <label for="age">อายุ:</label>
  <input type="number" id="age" name="age" min="1"
max="100" /><br>
  <label for="date">วันนัดหมาย:</label>
  <input type="date" id="date" name="date" /><br>
  <label for="color">สีที่ชอบ:</label>
  <input type="color" id="color" name="color" /><br>
  <input type="submit" value="ส่งข้อมูล" />
</form>
```



Input types ใหม่ที่สำคัญใน HTML5

	ประเภท	คำอธิบาย
-	email	ตรวจสอบรูปแบบอีเมลอัตโนมัติ
-	number	รับค่าตัวเลข พร้อม min/max
-	date, time	เลือกวันที่และเวลา
-	color	แสดงตัวเลือกสี
-	range	สไลด์บาร์สำหรับค่าเชิงตัวเลข
-	search	กล่องค้นหาที่ปรับแต่งพิเศษ

2.4 Multimedia: Audio / Video / Canvas

หนึ่งในความสามารถใหม่ที่สำคัญของ HTML5 คือการรองรับมัลติมีเดียโดยไม่ต้องใช้ plug-in เพิ่มเติมแบบใน HTML4 เช่น Flash หรือ Silverlight (Wikipedia Contributors, 2025)

(1) Audio และ Video

HTML5 เพิ่มแท็ก <audio> และ <video> สำหรับการเล่นเสียงและวิดีโอโดยตรงในเบราว์เซอร์

```
<h3>ตัวอย่างมัลติมีเดีย</h3>
<video controls width="320">
  <source src="sample.mp4" type="video/mp4" />
  เบราว์เซอร์ของคุณไม่รองรับวิดีโอ
</video>

<audio controls>
  <source src="sound.mp3" type="audio/mpeg" />
  เบราว์เซอร์ของคุณไม่รองรับเสียง
</audio>
```

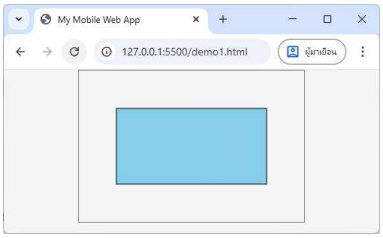
คุณสมบัติที่สำคัญ

- controls : แสดงปุ่มควบคุม (play, pause, volume)
- autoplay : เล่นอัตโนมัติเมื่อโหลดหน้า
- loop : เล่นซ้ำ
- source : ระบุไฟล์และชนิดของสื่อ

(2) Canvas API

<canvas> ช่วยให้วาดกราฟิกแบบไดนามิก เช่น กราฟ เกม หรือ animation ด้วย JavaScript

```
<canvas id="myCanvas" width="300" height="200" style="border:1px solid gray;"></canvas>
<script>
  const c = document.getElementById("myCanvas");
  const ctx = c.getContext("2d");
  ctx.fillStyle = "skyblue";
  ctx.fillRect(50, 50, 200, 100);
  ctx.strokeRect(50, 50, 200, 100);
</script>
```



โครงสร้างพื้นฐานของ HTML5 ได้ยกระดับความสามารถของเว็บจากเอกสารคงที่ไปสู่แอปพลิเคชันแบบโต้ตอบ โดยมุ่งเน้นทั้ง “ความหมายของข้อมูล (semantics)” และ “ประสบการณ์ของผู้ใช้ (user experience)” องค์ประกอบใหม่ เช่น semantic tags, input types, และ canvas ทำให้การพัฒนาเว็บมีความยืดหยุ่น และตอบโจทย์ยุค Mobile Web อย่างแท้จริง

3. การตกแต่งเอกสารด้วย CSS3

CSS (Cascading Style Sheets) คือภาษาที่ใช้สำหรับกำหนดรูปแบบและการนำเสนอของเอกสาร HTML หรือ XML โดยแยกส่วนของ “โครงสร้างข้อมูล” ออกจาก “การแสดงผล” เพื่อให้การออกแบบเว็บไซต์มีความยืดหยุ่นและดูแลรักษาได้ง่าย (W3C, 2023)

ในเวอร์ชัน CSS3 ได้เพิ่มคุณสมบัติใหม่ ๆ ที่ช่วยให้การออกแบบเว็บสมัยใหม่มีความสวยงาม โต้ตอบได้ (interactive) และรองรับอุปกรณ์หลากหลายมากยิ่งขึ้น เช่น Flexbox, Grid Layout, Animation และ Media Query

3.1 การใช้ Selector, Pseudo Class, Pseudo Element

(1) Selector

Selector คือเครื่องมือที่ใช้เลือกองค์ประกอบ HTML เพื่อกำหนดรูปแบบการแสดงผล เช่น เลือกตามชื่อแท็ก (class) หรือตัวระบุ (ID) (Mozilla Developer Network, n.d.)

```
<style>
  /* เลือกตามแท็ก */
  p { color: darkblue; }

  /* เลือกตาม class */
  .highlight { background-color: yellow; }

  /* เลือกตาม id */
  #main { border: 2px solid gray; }
</style>

<p id="main">ยินดีต้อนรับ</p>
<p class="highlight">นี่คือย่อหน้าที่ถูกเน้น</p>
```

(2) Pseudo-Class

Pseudo-Class ใช้กำหนดรูปแบบเฉพาะสถานะขององค์ประกอบ เช่น เมื่อเมาส์วางอยู่บนลิงก์ หรือ เมื่ออินพุตอยู่ในสถานะโฟกัส

```
<style>
  a:hover { color: red; }
  input:focus { border-color: dodgerblue; }
</style>
```

(3) Pseudo-Element

Pseudo-Element ใช้กำหนดรูปแบบส่วนใดส่วนหนึ่งขององค์ประกอบ เช่น อักษรตัวแรก หรือ การเพิ่มเนื้อหาก่อนและหลังแท็ก

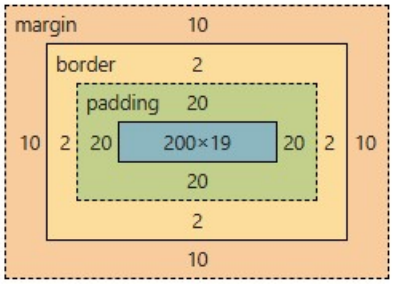
```
<style>
  p::first-letter { font-size: 1.5em; color: green; }
  p::after { content: " ✓"; color: gray; }
</style>
<p>ข้อความตัวอย่าง</p>
```

Pseudo-Classes และ Pseudo-Elements ช่วยเพิ่มความยืดหยุ่นในการออกแบบ โดยไม่ต้องแก้ไขโครงสร้าง HTML

3.2 Box Model, Flexbox, Grid Layout

(1) Box Model

ใน CSS ทุกองค์ประกอบจะถูกมองเป็น “กล่อง” ซึ่งประกอบด้วย content, padding, border และ margin โครงสร้างของ Box Model



ตัวอย่างโค้ด

```
<style>
.box {
  width: 200px;
  padding: 20px;
  border: 2px solid gray;
  margin: 10px;
}
</style>
```

(2) Flexbox

Flexbox หรือ Flexible Box Layout ช่วยจัดเรียงองค์ประกอบในแนวแกนหลัก (main axis) และแนวตั้งฉาก (cross axis) ได้อย่างยืดหยุ่น เหมาะกับการออกแบบ Responsive Layout (Mozilla Developer Network, n.d.)

ตัวอย่างโค้ด

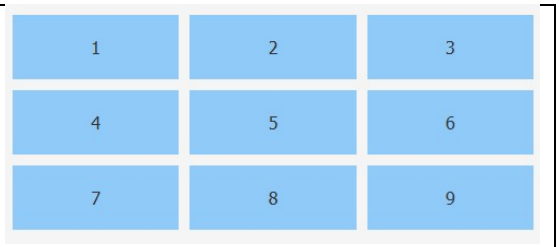
```
<style>
  .container {
    display: flex;
    justify-content: space-around;
    align-items: center;
```

```
    height: 200px;
    background-color: #e3f2fd;
  }
  .item {
    background-color: #2196f3;
    color: white;
    padding: 20px;
  }
</style>
<div class="container">
  <div class="item">กล่อง 1</div>
  <div class="item">กล่อง 2</div>
  <div class="item">กล่อง 3</div>
  <div class="item">กล่อง 4</div>
  <div class="item">กล่อง 5</div>
  <div class="item">กล่อง 6</div>
  <div class="item">กล่อง 7</div>
  <div class="item">กล่อง 8</div>
  <div class="item">กล่อง 9</div>
</div>
```

(3) Grid Layout

CSS Grid Layout เป็นระบบการจัดวางแบบ 2 มิติ (grid rows และ columns) ที่ให้การควบคุมโครงสร้างหน้าเว็บได้ละเอียดกว่าการใช้ Flexbox (W3C, 2023; CSS-Tricks, 2023)

```
<style>
  .grid {
    padding: 10px;
    display: grid;
    grid-template-columns: 1fr 1fr 1fr;
    gap: 10px;
  }
  .grid div {
    background-color: #90caf9;
    padding: 20px;
  }
</style>
<div class="grid">
  <div>1</div><div>2</div><div>3</div>
  <div>4</div><div>5</div><div>6</div>
  <div>7</div><div>8</div><div>9</div>
</div>
```



3.3 Animation และ Transition

CSS3 เพิ่มความสามารถด้านภาพเคลื่อนไหว โดยไม่ต้องใช้ JavaScript หรือ Flash (W3C, 2023)

(1) Transition

Transition ใช้สร้างเอฟเฟกต์เมื่อค่าของคุณสมบัติเปลี่ยนไป

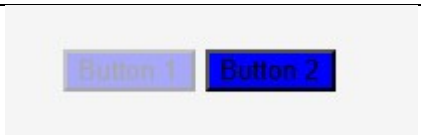
```
<style>
button {
  background-color: blue;
  transition: background-color 0.5s;
}
button:hover {
  background-color: red;
}
</style>
<button>Button 1</button>
<button>Button 2</button>
```

เมื่อใช้เมาส์เลื่อนไปที่ Button สีจะเปลี่ยนเป็นสีแดง

(2) Animation

Animation ใช้ keyframes ในการกำหนดลำดับการเปลี่ยนแปลงของคุณสมบัติ

```
<style>
@keyframes fadeIn {
  from { opacity: 0; }
  to { opacity: 1; }
}
.fade-in {
  animation: fadeIn 5s ease-in-out;
}
</style>
<button class="fade-in">Button 1</button>
<button >Button 2</button>
```



ปุ่ม Button 1 จะค่อยๆ ปรากฏขึ้น ในเวลา 5 วินาที

3.4 Media Query และ Responsive Layout

Media Query คือคุณสมบัติของ CSS3 ที่ใช้ตรวจสอบคุณลักษณะของอุปกรณ์ เช่น ความกว้างหน้าจอ และเลือกใช้สไตล์ที่เหมาะสม (Marcotte, 2010; BrowserStack Team, 2024)

```
<style>
 /* สไตล์พื้นฐาน (มือถือ) */
 body {
   background-color: #fff;
 }
 /* สำหรับหน้าจอกว้างกว่า 500px */
 @media (min-width: 500px) {
   body {
     background-color: #e4d910;
   }
 }
</style>
```

หน้าเว็บจะแสดงสีตามขนาดของหน้าจอ

แนวคิด Responsive Layout มุ่งหมายให้หน้าเว็บ “ตอบสนอง (respond)” ต่อขนาดหน้าจอและอุปกรณ์ โดยผสาน Media Query กับ Flexbox หรือ Grid เพื่อให้โครงสร้างหน้าเว็บยืดหยุ่น

CSS3 คือหัวใจของการออกแบบเว็บไซต์สมัยใหม่ ที่ช่วยให้การนำเสนอเนื้อหามีความยืดหยุ่น สวยงาม และเหมาะสมกับทุกอุปกรณ์ คุณสมบัติหลัก เช่น Selector, Pseudo-Class, Flexbox, Grid, Animation และ Media Query ช่วยให้ผู้พัฒนาสามารถสร้าง Responsive และ Interactive Web ได้อย่างสมบูรณ์