**เปรียบเทียบ Utility Class กับ CSS ปกติ**

**1. Utility Class (เช่น Tailwind CSS)**

**แนวคิด:**

* ใช้คลาสที่กำหนดล่วงหน้า (เช่น bg-blue-500, text-center, p-4) สำหรับการจัดรูปแบบ
* ทุกอย่างถูกจัดการผ่าน HTML โดยตรง
* ไม่ต้องเขียนไฟล์ CSS แยก

**ข้อดี:**

* **พัฒนาเร็ว:** ไม่ต้องเขียน CSS เอง เพียงแค่เพิ่มคลาสใน HTML
* **สม่ำเสมอ:** ได้ดีไซน์ที่คงที่ตลอดทั้งโปรเจกต์
* **ลด CSS ซ้ำซ้อน:** ไม่ต้องสร้าง CSS ที่มีคุณสมบัติซ้ำกัน
* **คุมขนาดไฟล์ได้ดี:** เมื่อมีการ build ไฟล์จะรวมเฉพาะคลาสที่ใช้งานจริง

**ข้อเสีย:**

* **อ่านยาก:** HTML จะมีคลาสจำนวนมาก ทำให้อ่านโค้ดยากขึ้น
* **ปรับแต่งยาก:** การทำดีไซน์เฉพาะเจาะจงบางครั้งต้องเพิ่ม CSS เอง
* **Dependency:** ขึ้นอยู่กับเฟรมเวิร์ก ทำให้ต้องเรียนรู้โครงสร้างใหม่

**ตัวอย่าง (Utility Class):**

<div class="bg-blue-500 text-white p-4 rounded-lg">

ปุ่มตัวอย่าง

</div>

**2. CSS ปกติ (Vanilla CSS)**

**แนวคิด:**

* เขียน CSS แยกจาก HTML
* กำหนดสไตล์ผ่าน Selector เช่น .btn {} หรือ h1 {}

**ข้อดี:**

* **โค้ดสะอาด:** HTML ไม่รกด้วยคลาสมากมาย
* **ปรับแต่งง่าย:** มีความยืดหยุ่นสูง สามารถเขียน CSS ตามที่ต้องการ
* **แยกส่วนดี:** HTML และ CSS ถูกแยกจากกันทำให้อ่านง่าย

**ข้อเสีย:**

* **เขียน CSS ซ้ำ:** ต้องเขียนสไตล์คล้ายกันหลายครั้ง
* **ไฟล์ใหญ่:** CSS มีโอกาสโตเร็วและจัดการยากหากไม่มีการวางแผนที่ดี
* **เวลาในการพัฒนา:** ต้องใช้เวลาในการเขียนและทดสอบ

**ตัวอย่าง (CSS ปกติ):**  
**HTML:**

<div class="btn">ปุ่มตัวอย่าง</div>

**CSS:**

.btn {

background-color: #3b82f6;

color: white;

padding: 1rem;

border-radius: 0.5rem;

}

**สรุปการใช้งาน**

* **Utility Class (เช่น Tailwind):** เหมาะสำหรับโปรเจกต์ที่ต้องการความเร็วและดีไซน์ที่สม่ำเสมอ
* **CSS ปกติ:** เหมาะกับโปรเจกต์ที่ต้องการการควบคุมสูงและมีดีไซน์เฉพาะเจาะจง

**การเลือกใช้งานขึ้นอยู่กับ:**

* **ขนาดโปรเจกต์**
* **ความเร็วในการพัฒนา**
* **ความต้องการในการปรับแต่งดีไซน์**

66160157 น.ส.ศิริลักษณ์ ชุลีพันธุ์