## เว็บไซต์ทำงานอย่างไร

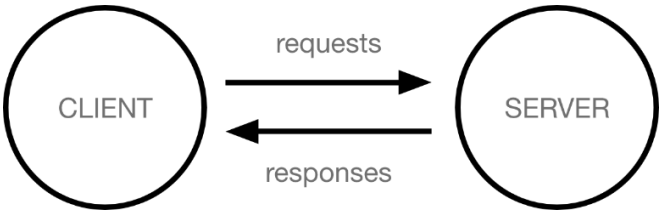
World Wide Web (WWW) หรือเว็บเป็นระบบที่ใช้ในการเชื่อมต่อข้อมูลผ่านทางอินเทอร์เน็ต โดยใช้โปรโตคอล HTTP (Hypertext Transfer Protocol) ซึ่งเป็นโปรโตคอลสื่อสารระหว่างเว็บเซิร์ฟเวอร์และเบราว์เซอร์ การทำงานของ WWW ประกอบด้วยส่วนประกอบหลัก ๆ ดังต่อไปนี้:

1. **เว็บเซิร์ฟเวอร์ (Web Server):** เป็นเครื่องคอมพิวเตอร์หรือเซิร์ฟเวอร์ที่เก็บข้อมูลและไฟล์ของเว็บไซต์ ซึ่งมักใช้โปรแกรมเซิร์ฟเวอร์เช่น Apache, Nginx เป็นต้น เมื่อมีคำขอ (request) จากเบราว์เซอร์ผ่าน HTTP แล้วเว็บเซิร์ฟเวอร์จะตอบกลับด้วยข้อมูลที่ต้องการ
2. **เบราว์เซอร์ (Web Browser):** เป็นโปรแกรมที่ใช้ในการเข้าถึงและแสดงผลข้อมูลจากเว็บไซต์ โดยส่วนใหญ่เบราว์เซอร์ที่รู้จักกันมากที่สุดคือ Google Chrome, Mozilla Firefox, Safari, Microsoft Edge ฯลฯ ผู้ใช้งานสามารถใช้เบราว์เซอร์ในการพิมพ์ URL (Uniform Resource Locator) เพื่อเข้าถึงเว็บไซต์ที่ต้องการ
3. **HTML (Hypertext Markup Language):** เป็นภาษาที่ใช้ในการเขียนโค้ดของหน้าเว็บไซต์ ซึ่งใช้สำหรับกำหนดโครงสร้างและรูปแบบของเนื้อหาบนหน้าเว็บ
4. **HTTP (Hypertext Transfer Protocol):** เป็นโปรโตคอลสื่อสารที่ใช้ในการส่งข้อมูลระหว่างเว็บเซิร์ฟเวอร์และเบราว์เซอร์ โดยเมื่อผู้ใช้งานเปิดหน้าเว็บในเบราว์เซอร์ เบราว์เซอร์จะส่งคำขอ (request) ไปยังเว็บเซิร์ฟเวอร์และเว็บเซิร์ฟเวอร์จะตอบกลับด้วยข้อมูล (response) ซึ่งส่วนใหญ่จะเป็น HTML และแสดงผลบนเบราว์เซอร์
5. **URL (Uniform Resource Locator):** เป็นที่อยู่ของเอกสารหรือไฟล์บนเว็บ ซึ่งใช้ในการระบุที่อยู่ของเว็บไซต์ที่ผู้ใช้งานต้องการเข้าถึง

โดยทั่วไปแล้ว การทำงานของ WWW จะเริ่มต้นจากผู้ใช้งานเปิดเบราว์เซอร์และพิมพ์ URL หรือคลิกลิงก์เพื่อเข้าถึงเว็บไซต์ที่ต้องการ จากนั้นเบราว์เซอร์จะส่งคำขอไปยังเว็บเซิร์ฟเวอร์ผ่าน HTTP และเว็บเซิร์ฟเวอร์จะตอบกลับด้วยข้อมูลที่ต้องการ เบราว์เซอร์จะแปลงข้อมูล HTML เป็นหน้าเว็บที่ผู้ใช้งานเห็นได้ และแสดงผลบนหน้าจอของผู้ใช้งาน

## Client และ Server

ในระบบเว็บเทคโนโลยี การทำงานระหว่าง Client และ Server จะเกิดขึ้นเป็นลูกศรแบบสองทิศทาง (two-way communication) โดย Client จะส่งคำขอไปยัง Server และ Server จะตอบกลับด้วยข้อมูลที่ต้องการ ข้อมูลจะถูกส่งผ่านโปรโตคอลที่ถูกกำหนดไว้ เช่น HTTP, HTTPS และการสื่อสารระหว่าง Client และ Server จะเกิดขึ้นผ่านการใช้งานเน็ตเวิร์ก อย่างเช่น TCP/IP หรือโปรโตคอลอื่น ๆ ที่เกี่ยวข้อง




### Client

Client เป็นส่วนที่อยู่ฝั่งผู้ใช้งาน โดยทั่วไปจะเป็นเบราว์เซอร์ที่ใช้ในการเข้าถึงเว็บไซต์ หรือแอปพลิเคชันที่ใช้งานผ่านเว็บ หน้าที่ของ Client รวมถึง:

- ส่งคำขอ (request) ไปยัง Server เพื่อขอข้อมูลหรือบริการต่าง ๆ เช่น เว็บเพจ, ไฟล์, ข้อมูลจากฐานข้อมูล ฯลฯ
- แสดงผลข้อมูลที่ได้รับจาก Server บนเว็บเบราว์เซอร์
- ประมวลผลและจัดการกับเหตุการณ์ที่เกิดขึ้นในส่วนของผู้ใช้งาน เช่น การคลิกลิงก์, การกรอกแบบฟอร์ม เป็นต้น

### Server

Server เป็นส่วนที่อยู่ฝั่งเซิร์ฟเวอร์ หรือคอมพิวเตอร์ที่ทำหน้าที่ให้บริการแก่ Client โดยตลอดเวลา หน้าที่หลักของ Server รวมถึง:

- รอรับคำขอ (request) จาก Client ที่เข้ามาผ่านทางเน็ตเวิร์ก
- ประมวลผลคำขอและดำเนินการตามคำขอต่าง ๆ เช่น การเข้าถึงข้อมูลจากฐานข้อมูล, การสร้างหน้าเว็บแบบพิเศษ เป็นต้น
- ส่งคำตอบ (response) กลับไปยัง Client ซึ่งอาจประกอบด้วยข้อมูลที่ขอมาหรือข้อความแจ้งเตือนอื่น ๆ
- จัดเก็บและบริหารจัดการข้อมูลต่าง ๆ ที่เกี่ยวข้องกับเว็บไซต์ หรือแอปพลิเคชันนั้น ๆ

## จริง ๆ แล้วมันทำงานยังไง

เมื่อคุณพิมพ์ที่อยู่เว็บลงในเบราว์เซอร์ของคุณ (เพื่อให้เข้าใจง่าย ให้เปรียบเสมือนการเดินไปยังร้านค้า):

1. เบราว์เซอร์จะติดต่อไปยังเซิร์ฟเวอร์ DNS เพื่อค้นหาที่อยู่จริงของเซิร์ฟเวอร์ที่เก็บเว็บไซต์นั้นอยู่ (คุณค้นหาที่อยู่ของร้านค้า)
2. เบราว์เซอร์จะส่งข้อความ (Request) ผ่านโปรโตคอล HTTP ไปยังเซิร์ฟเวอร์ ขอให้เซิร์ฟเวอร์ส่งสำเนาของเว็บไซต์ให้กับไคลเอนต์ (คุณไปที่ร้านค้าและสั่งสินค้า ในที่นี้ สินค้าคือ หน้าเว็บไซต์)
3. หากเซิร์ฟเวอร์อนุมัติคำขอของไคลเอนต์ เซิร์ฟเวอร์จะส่งข้อความ "200 OK" กลับไปยังไคลเอนต์ ซึ่งหมายความว่า "แน่นอนว่าคุณสามารถดูเว็บไซต์นั้นได้! นี่คือเว็บไซต์" และเริ่มส่งไฟล์ของเว็บไซต์ให้กับเบราว์เซอร์เป็นชุดข้อมูลเล็ก ๆ ที่เรียกว่าแพ็กเก็ตข้อมูล (ร้านค้าจะให้สินค้าให้คุณ และคุณก็นำสินค้ากลับมาบ้านของคุณ)
4. เบราว์เซอร์จะประกอบ (Assemble) และแสดงเว็บเพจเต็มรูปแบบให้คุณเห็น (แกะดูสินค้ามาที่บ้านของคุณ – ได้สินค้าใหม่ที่สวยงามและน่าตื่นเต้น)

### DNS (Domain Name System)

DNS ทำหน้าที่แปลงชื่อโดเมน (Domain Name) ให้เป็นที่อยู่ IP (IP Address) ที่เซิร์ฟเวอร์สามารถเข้าถึงได้ โดยทำงานตามขั้นตอนดังนี้:

1. เมื่อผู้ใช้งานป้อนชื่อโดเมนลงในเบราว์เซอร์ เช่น www.example.com เบราว์เซอร์จะส่งคำขอ DNS lookup ไปยังเซิร์ฟเวอร์ DNS ที่ตั้งค่าไว้เพื่อการแปลงชื่อโดเมน
2. เบราว์เซอร์จะส่งคำขอ DNS lookup ไปยังเซิร์ฟเวอร์ DNS ที่ถูกกำหนดให้ในการติดต่อเป็นไปตามลำดับ ซึ่งอาจเป็น DNS resolver ภายในองค์กรหรือเซิร์ฟเวอร์ DNS สาธารณะที่ให้บริการ
3. เซิร์ฟเวอร์ DNS ที่ได้รับคำขอจะทำการค้นหา Name Server ของโดเมนที่ถูกต้อง โดยเรียกใช้งาน DNS Root Servers ที่มีทั้งหมด 13 เซิร์ฟเวอร์ ซึ่งจะได้รับข้อมูลเซิร์ฟเวอร์ DNS ระดับสูงของโดเมนนั้น ๆ

4. เซิร์ฟเวอร์ DNS จะส่งคำขอแบบ recursive ไปยังเซิร์ฟเวอร์ DNS ระดับสูงของโดเมนนั้น ๆ เพื่อค้นหาข้อมูลเกี่ยวกับโดเมน เซิร์ฟเวอร์ DNS ระดับสูงจะทำการวิเคราะห์และติดตามรายการเซิร์ฟเวอร์ DNS ต่อไป เพื่อค้นหาข้อมูลที่เกี่ยวข้องกับโดเมน

5. เซิร์ฟเวอร์ DNS ระดับสูงจะส่งคำตอบ DNS response กลับไปยังเซิร์ฟเวอร์ DNS ที่ส่งคำขอ และคำตอบจะรวมถึงที่อยู่ IP สำหรับโดเมนที่ต้องการ

6. เซิร์ฟเวอร์ DNS ที่ส่งคำขอจะได้รับคำตอบ DNS response จากเซิร์ฟเวอร์ DNS ระดับสูง และจะส่งคำตอบกลับไปยังเบราว์เซอร์ที่ได้ส่งคำขอ DNS lookup

7. เบราว์เซอร์จะได้รับที่อยู่ IP จากคำตอบ DNS response และจะใช้ที่อยู่ IP นี้ในการเชื่อมต่อกับเว็บไซต์ที่ผู้ใช้งานต้องการเข้าถึง จากนั้นเบราว์เซอร์จะแสดงผลเว็บไซต์ให้กับผู้ใช้งาน

โดยการทำงานของ DNS ช่วยให้ผู้ใช้งานสามารถเข้าถึงเว็บไซต์โดยใช้ชื่อโดเมนที่จำเป็นแทนที่จะต้องจำที่อยู่ IP ของเว็บไซต์นั้น ๆ และช่วยให้เกิดการเชื่อมต่อและสื่อสารระหว่างเบราว์เซอร์และเซิร์ฟเวอร์ได้อย่างราบรื่น

## โปรโตคอล (Protocol)

โปรโตคอล เป็นชุดกฎเกณฑ์และกฎเกณฑ์ที่ใช้ในการสื่อสารระหว่างระบบคอมพิวเตอร์ เพื่อให้ข้อมูลและข้อความสามารถถูกส่งและรับรู้ได้อย่างถูกต้องและเป็นระเบียบ โปรโตคอลรับผิดชอบในการกำหนดรูปแบบข้อมูลที่จะถูกส่ง, โครงสร้างข้อความ, ขั้นตอนวิธีการสื่อสาร และวิธีการจัดการข้อผิดพลาดที่อาจเกิดขึ้นในการสื่อสาร

สำหรับเว็บเทคโนโลยีสำคัญ โปรโตคอลที่มีบทบาทสำคัญได้แก่:

1. HTTP (Hypertext Transfer Protocol): เป็นโปรโตคอลที่ใช้ในการสื่อสารระหว่างเบราว์เซอร์และเซิร์ฟเวอร์ในรูปแบบของข้อความแบบไฮเปอร์เท็กซ์ (Hypertext) เป็นพื้นฐานในการส่งคำขอและรับคำตอบเว็บ และใช้ในการแสดงผลเว็บไซต์ที่เราเห็นบนเบราว์เซอร์

2. HTTPS (Hypertext Transfer Protocol Secure): เป็นรูปแบบของ HTTP ที่มีการเพิ่มความปลอดภัย โดยใช้การเข้ารหัสข้อมูลที่ถูกส่งระหว่างเบราว์เซอร์และเซิร์ฟเวอร์ โปรโตคอลนี้ใช้ในการสื่อสารที่เกี่ยวกับข้อมูลที่ละเอียดและที่ต้องการความปลอดภัย เช่น การทำธุรกรรมทางการเงินออนไลน์

3. TCP/IP (Transmission Control Protocol/Internet Protocol): เป็นชุดโปรโตคอลที่ใช้ในการสื่อสารระหว่างคอมพิวเตอร์ในเครือข่ายอินเตอร์เน็ต โปรโตคอล TCP/IP เกี่ยวข้องกับการแบ่งข้อมูลเป็นแพ็กเก็ต เชื่อมต่อเครือข่าย ติดตามเส้นทางของแพ็กเก็ต เป็นต้น

4. DNS (Domain Name System): เป็นโปรโตคอลที่ใช้ในการแปลงชื่อโดเมนเป็นที่อยู่ IP ที่เซิร์ฟเวอร์สามารถเข้าถึงได้ เมื่อเราพิมพ์ที่อยู่เว็บในเบราว์เซอร์ เป็นบริการที่สำคัญในการเชื่อมต่อไปยังเว็บไซต์ที่ต้องการ

โปรโตคอลเหล่านี้ทำงานร่วมกันเพื่อให้การสื่อสารระหว่างเบราว์เซอร์และเซิร์ฟเวอร์เป็นไปอย่างราบรื่นและปลอดภัยในเว็บเทคโนโลยี

## Developers must know

สำหรับนักพัฒนาเว็บไซต์เพื่อพัฒนาเว็บแอปพลิเคชันหรือเว็บเซอร์วิส (Web Application or Web Services) ควรมีความรู้และทักษะต่อไปนี้:

1. **HTML (Hypertext Markup Language):** เป็นภาษาหลักในการสร้างโครงสร้างและเนื้อหาของหน้าเว็บไซต์ จะใช้ในการสร้างโครงสร้างของเว็บและเพื่อแสดงข้อมูลให้กับผู้ใช้งาน

2. **CSS (Cascading Style Sheets):** เป็นภาษาที่ใช้ในการกำหนดรูปแบบและสไตล์การแสดงผลของหน้าเว็บ จะใช้ในการจัดรูปแบบเพื่อทำให้เว็บไซต์ดูสวยงามและมีการจัดวางเนื้อหาอย่างเหมาะสม

3. **JavaScript:** เป็นภาษาโปรแกรมมิ่งที่ใช้ในการเพิ่มความประสิทธิภาพและความประสบความสำเร็จในการสร้างเว็บแอปพลิเคชัน สามารถใช้ในการจัดการเหตุการณ์ที่เกิดขึ้นบนเว็บไซต์ การเรียกใช้งาน API, การจัดการข้อมูลแบบอื่น ๆ และอื่น ๆ

4. **เครื่องมือพัฒนาเว็บ:** สำหรับการพัฒนาเว็บไซต์ จำเป็นต้องใช้เครื่องมือต่าง ๆ เช่น Text Editor, Integrated Development Environment (IDE) เช่น Visual Studio Code, Sublime Text, หรือ WebStorm เพื่อเขียนแก้ไขและบริหารจัดการโค้ด เครื่องมือตรวจสอบข้อผิดพลาด (Debugger) เพื่อตรวจสอบและแก้ไขข้อผิดพลาดที่เกิดขึ้น และเครื่องมืออื่น ๆ ที่ช่วยในกระบวนการพัฒนาเว็บ

5. **APIs (Application Programming Interfaces):** ในกรณีที่ต้องการเชื่อมต่อกับบริการหรือแหล่งข้อมูลภายนอก เช่น การใช้งาน Google Maps, การเรียกใช้ข้อมูลจากฐานข้อมูล หรือการติดต่อกับบริการอื่น ๆ ที่ให้บริการผ่าน APIs สำหรับการพัฒนาเว็บไซต์จำเป็นต้องเข้าใจและใช้ APIs ให้ถูกต้อง

6. **การจัดการฐานข้อมูล:** ในกรณีที่ต้องการเก็บข้อมูลหรือดึงข้อมูลจากฐานข้อมูล เช่น MySQL, PostgreSQL, MongoDB เป็นต้น นักพัฒนาจะต้องมีความรู้ในการออกแบบและใช้งานฐานข้อมูลในเว็บไซต์

นอกจากนี้ยังควรมีความรู้เพิ่มเติมเกี่ยวกับการทำงานของเว็บเซิร์ฟเวอร์ เรื่องการรักษาความปลอดภัย (Security) เช่น HTTPS, Cross-Site Scripting (XSS), Cross-Site Request Forgery (CSRF) และเครื่องมือในการทดสอบและประสิทธิภาพของเว็บไซต์ เช่น Load Testing, Performance Optimization ฯลฯ

นอกจากภาษา ความรู้ทักษะด้านการออกแบบ UI/UX และการทำ Responsive Web Design ยังมีความสำคัญในการพัฒนาเว็บไซต์ที่ดีและมีประสิทธิภาพ

## ภาษา HTML

มีหลาย tag ที่สำคัญ และสามารถแบ่งเป็นกลุ่มต่าง ๆ ได้ดังนี้:

1. **Document Structure Tags**: ใช้สำหรับการกำหนดโครงสร้างของเอกสาร HTML
    - **<!DOCTYPE html>**: กำหนดประเภทของเอกสาร HTML
    - **<html>**: กำหนดเริ่มต้นและสิ้นสุดของเอกสาร HTML
    - **<head>**: บรรจุข้อมูลเมตาและลิงก์ไปยังสไตล์ชีตและสคริปต์ JavaScript
    - **<title>**: กำหนดชื่อเรื่องของเอกสาร HTML ที่แสดงในแท็บของเบราว์เซอร์
    - **<body>**: บรรจุเนื้อหาหลักของเว็บเพจ
2. **Text Formatting Tags**: ใช้สำหรับการจัดรูปแบบข้อความ
    - **<h1>** ถึง **<h6>**: กำหนดหัวข้อขนาดต่าง ๆ
    - **<p>**: กำหนดย่อหน้า
    - **<b>**: ทำให้ข้อความเป็นตัวหนา
    - **<i>**: ทำให้ข้อความเป็นตัวเอียง
    - ****: ทำให้ข้อความเป็นตัวหนา
    - ****: ทำให้ข้อความเป็นตัวเอียง
    - **<small>**: ทำให้ข้อความเล็กลง
    - **<del>**: ขีดฆ่าข้อความ
    - **<ins>**: ขีดเส้นใต้ข้อความ
    - **<sub>**: ทำให้ข้อความเป็นตัวห้อย
    - **<sup>**: ทำให้ข้อความเป็นตัวยก
3. **Link and Image Tags**: ใช้สำหรับการสร้างลิงก์และแทรกภาพ
    - **<a>**: สร้างลิงก์
    - **<img>**: แทรกภาพ

4. List Tags: ใช้สำหรับการสร้างรายการ
    - <ul>: สร้างรายการที่ไม่มีลำดับ (unordered list)
    - <ol>: สร้างรายการที่มีลำดับ (ordered list)
    - <li>: กำหนดรายการใน <ul> หรือ <ol>
5. Table Tags: ใช้สำหรับการสร้างตาราง
    - <table>: สร้างตาราง
    - <tr>: สร้างแถวในตาราง
    - <td>: สร้างเซลล์ในแถว
    - <th>: สร้างเซลล์หัวข้อในแถว
6. Form Tags: ใช้สำหรับการสร้างฟอร์ม
    - <form>: สร้างฟอร์ม
    - <input>: สร้างอินพุตต่าง ๆ เช่น ข้อความ, ปุ่ม, ช่องทำเครื่องหมาย, และอื่น ๆ
    - <textarea>: สร้างพื้นที่ให้ผู้ใช้ป้อนข้อความ
    - <button>: สร้างปุ่ม
    - <select>: สร้างเมนูแบบเลื่อนลง
    - <option>: กำหนดตัวเลือกใน <select>
7. Semantic Tags: ใช้สำหรับการกำหนดความหมายของเนื้อหา
    - <header>: กำหนดส่วนหัวของเอกสารหรือส่วนหนึ่งของเอกสาร
    - <footer>: กำหนดส่วนท้ายของเอกสารหรือส่วนหนึ่งของเอกสาร
    - <main>: กำหนดเนื้อหาหลักของเอกสาร
    - <section>: กำหนดส่วนหนึ่งของเอกสาร
    - <article>: กำหนดบทความ
    - <aside>: กำหนดเนื้อหาที่เกี่ยวข้องกับเนื้อหาหลักของเอกสาร

- **<nav>**: กำหนดการนำทางลิงก์

## CSS (Cascading Style Sheets)

มีหลายคำสั่งที่สำคัญ และสามารถแบ่งเป็นกลุ่มต่าง ๆ ได้ดังนี้:

1. **Text and Font Styles**: ใช้สำหรับการจัดรูปแบบข้อความและฟอนต์
    - **color**: กำหนดสีของข้อความ
    - **font-family**: กำหนดฟอนต์ของข้อความ
    - **font-size**: กำหนดขนาดของฟอนต์
    - **font-weight**: กำหนดความหนาของฟอนต์
    - **text-align**: กำหนดการจัดวางข้อความ (left, right, center, justify)
    - **text-decoration**: กำหนดการตกแต่งข้อความ (none, underline, line-through, overline)
2. **Box Model**: ใช้สำหรับการจัดรูปแบบของกล่องที่รอบรัดองค์ประกอบ HTML
    - **margin**: กำหนดระยะห่างของขอบภายนอก
    - **padding**: กำหนดระยะห่างของขอบภายใน
    - **border**: กำหนดขอบ
    - **width** and **height**: กำหนดความกว้างและความสูง
3. **Layout Properties**: ใช้สำหรับการจัดรูปแบบและการวางตำแหน่งขององค์ประกอบ
    - **display**: กำหนดวิธีการแสดงองค์ประกอบ (block, inline, none, flex, grid)
    - **position**: กำหนดวิธีการวางตำแหน่ง (static, relative, absolute, fixed, sticky)
    - **top**, **bottom**, **left**, **right**: กำหนดตำแหน่งเมื่อใช้กับ **position**
    - **float**: กำหนดวิธีการลอยขององค์ประกอบ
    - **clear**: กำหนดวิธีการล้างการลอย
4. **Background and Color**: ใช้สำหรับการจัดรูปแบบพื้นหลังและสี
    - **background-color**: กำหนดสีพื้นหลัง

- **background-image**: กำหนดรูปภาพพื้นหลัง
- **background-repeat**: กำหนดวิธีการทำซ้ำรูปภาพพื้นหลัง
- **background-position**: กำหนดตำแหน่งของรูปภาพพื้นหลัง
- **background-size**: กำหนดขนาดของรูปภาพพื้นหลัง

5. **List and Table Styles**: ใช้สำหรับการจัดรูปแบบรายการและตาราง
   - **list-style-type**: กำหนดรูปแบบของเครื่องหมายรายการ
   - **list-style-position**: กำหนดตำแหน่งของเครื่องหมายรายการ
   - **border-collapse**: กำหนดวิธีการรวมขอบของตาราง
   - **border-spacing**: กำหนดระยะห่างระหว่างขอบของตาราง
6. **Pseudo-classes and Pseudo-elements**: ใช้สำหรับการกำหนดรูปแบบในสถานะหรือส่วนที่เฉพาะเจาะจงขององค์ประกอบ
   - **:hover, :active, :visited, :focus**: กำหนดรูปแบบของลิงก์หรือองค์ประกอบอื่น ๆ ในสถานะที่แตกต่างกัน
   - **::before, ::after**: ใส่เนื้อหาก่อนหน้าหรือหลังจากองค์ประกอบ