Think Beyond Develop

Think Beyond

#โมเดิร์น JavaScript

เก่งได้ใน 30 วัน

ฉบับ
Node.js
+
MongoDB

- พื้นฐาน JavaScript ครบทุกแง่มุม สำหรับนักเรียน นักศึกษา และโปรแกรมเมอร์
- การใช้ JavaScript จัดการกับ DOM
- สร้างเว็บแอปพลิเคชันฝั่งเซิร์ฟเวอร์ด้วย Node.js
- พื้นฐานการใช้งานฐานข้อมูล MongoDB พร้อมตัวอย่างแอปพลิเคชัน
- วิธีใช้งานแพ็กเกจยอดนิยม เช่น express, mongoose, jsonwebtoken, ejs และอื่น ๆ

SCAN FREE

ไฟล์ประกอบหนังสือ

จีราวุธ วารินทร์

จัดจำหน่ายโดย IDC PREMIER

สารบัญ

บทที่ 1
แนะนำ JavaScript

JavaScript (จาวาสคริปต์) เป็นภาษาสคริปต์ที่ใช้กับเว็บเบราว์เซอร์ นักพัฒนาสามารถนำ JavaScript มามาสร้างเว็บแอปพลิเคชันทั้งในฝั่งไคลเอนต์และฝั่งเซิร์ฟเวอร์

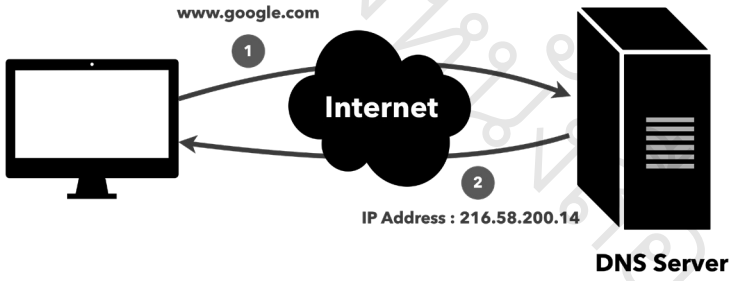
ในบทแรกนี้จะมาทำความรู้จักกับภาษา JavaScript และวิธีการทดสอบโค้ด JavaScript ผ่านทางอินเทอร์เน็ต วิธีนี้มีข้อดีคือ ไม่ต้องติดตั้งอะไรลงไปในเครื่องคอมพิวเตอร์ก็สามารถทดสอบ หรือเรียนรู้การใช้ JavaScript ได้ทันที

ทบทวนเกี่ยวกับอินเทอร์เน็ต (Internet)

อินเทอร์เน็ต คือ คอมพิวเตอร์ที่เชื่อมต่อกันเป็นระบบเครือข่ายขนาดใหญ่ที่เชื่อมต่อกันทั้งโลก โดยเรียกคอมพิวเตอร์ที่ทำหน้าที่ให้บริการข้อมูล หรือบริการด้านต่าง ๆ บนอินเทอร์เน็ตว่า Server (เซิร์ฟเวอร์) และเรียกคอมพิวเตอร์ที่เชื่อมต่อไปยังอินเทอร์เน็ต และใช้บริการต่าง ๆ จากอินเทอร์เน็ต ว่า Client (ไคลเอนต์)

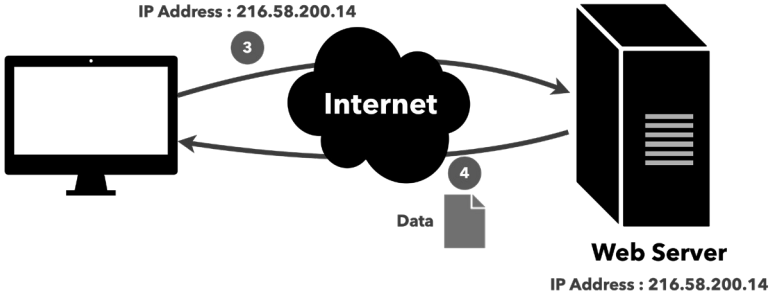
คอมพิวเตอร์และอุปกรณ์ต่าง ๆ ที่เชื่อมต่ออินเทอร์เน็ต จะมีหมายเลข IP Address ที่ไม่ซ้ำกัน การสื่อสารระหว่างคอมพิวเตอร์ที่อยู่ในอินเทอร์เน็ต จะใช้ IP Address ระบุตำแหน่งของผู้รับและผู้ส่งข้อมูลในอินเทอร์เน็ต

เมื่อผู้ใช้ต้องการเปิดดูหน้าเว็บเพจที่อยู่ในอินเทอร์เน็ต จะกรอก URL เช่น thinkbeyondbook.com เพื่อส่งที่อยู่ที่ต้องการติดต่อไปให้กับ DNS Server (Domain Name System Server) เพื่อแปลง URL ให้กลายเป็นหมายเลข IP Address และส่งกลับมายังเบราว์เซอร์



▲ รูปแสดงส่งข้อมูลระหว่างเบราว์เซอร์ และ DNS Server

หลังจากที่เบราว์เซอร์ทราบหมายเลข IP Address ก็จะส่งคำร้องขอไปยังเว็บเซิร์ฟเวอร์ตามหมายเลข IP Address ดังกล่าว เซิร์ฟเวอร์ก็จะส่งข้อมูลกลับมาแสดงผลยังเบราว์เซอร์ตามต้องการ



▲ รูปแสดงส่งข้อมูลระหว่างเบราว์เซอร์ และ Web Server

- JavaScript คือ ไฟล์ซึ่งทำให้เว็บเพจสามารถโต้ตอบกับผู้ใช้ได้ หรือทำให้เว็บเพจมีความสามารถบางอย่างเพิ่มขึ้นมา ซึ่งไฟล์ JavaScript จะมีนามสกุลไฟล์เป็น .js

การเขียนโค้ด JavaScript จะมีความสัมพันธ์โดยตรงกับโค้ด HTML และ CSS ที่อยู่ในเว็บเพจ เช่น เขียนโค้ด JavaScript เพื่อแสดงหรือซ่อนเนื้อหาที่อยู่ใน HTML หรือเขียนโค้ด JavaScript เพื่อเลือกใช้งาน CSS สำหรับปรับแต่งหน้าตาเว็บเพจในแบบที่ต้องการ

ภาษา JavaScript

JavaScript เป็นภาษาคอมพิวเตอร์ ที่ถูกออกแบบมาให้ทำงานบนเบราว์เซอร์ จุดประสงค์เพื่อทำให้เว็บเพจสามารถโต้ตอบกับผู้ใช้ได้ หรือมีความสามารถบางอย่างเพิ่มขึ้น เช่น สามารถใช้ JavaScript เพื่อขยายรูปภาพเมื่อผู้ใช้เลื่อนเมาส์ไปที่รูปภาพ, ใช้ JavaScript เพื่อสร้างแอนิเมชันเลื่อนข้อความจากซ้ายไปขวา, ใช้ JavaScript เพื่อนำเนื้อหาใหม่มาแทนที่เนื้อหาเดิมโดยไม่ต้องโหลดเว็บเพจใหม่ เป็นต้น

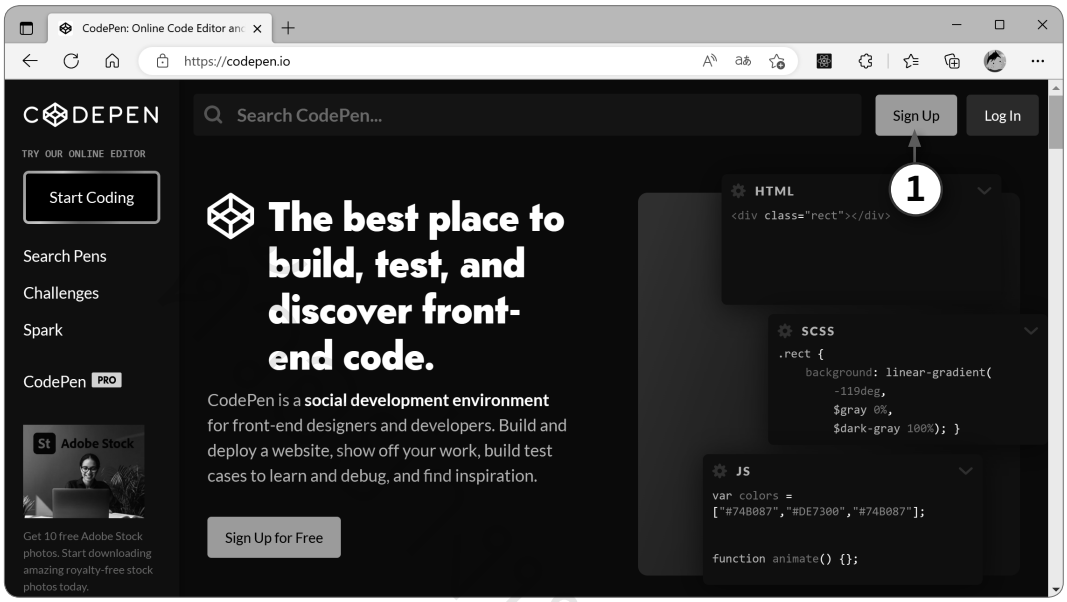
JavaScript เป็นภาษาคอมพิวเตอร์ที่ออกแบบมาสำหรับใช้กับเว็บเพจ ดังนั้น โค้ด JavaScript จึงถูกรันเพื่อใช้งานบนเบราว์เซอร์เป็นหลัก นอกจากนั้น JavaScript ยังสามารถใช้กับสภาพแวดล้อมอื่น ๆ ที่ไม่ใช่เบราว์เซอร์ได้อีกด้วย เช่น ใช้ JavaScript กับ Node.js และ Apache CouchDB เป็นต้น

▲ รูปแสดงตัวอย่างโค้ดคำสั่ง JavaScript

ลงทะเบียนกับ https://codepen.io

ก่อนทดสอบโค้ด JavaScript กับ CodePen จะต้องสมัครสมาชิกก่อน เพื่อให้มีโปรไฟล์สำหรับเก็บโค้ดที่ได้เขียนไว้ และสามารถแชร์โค้ดไปให้สมาชิกคนอื่น ๆ สามารถนำโค้ดไปใช้งานได้

1. เปิดเบราว์เซอร์ และไปยังเว็บไซต์ https://codepen.io แล้วคลิกปุ่ม Sign Up เพื่อสมัครสมาชิก



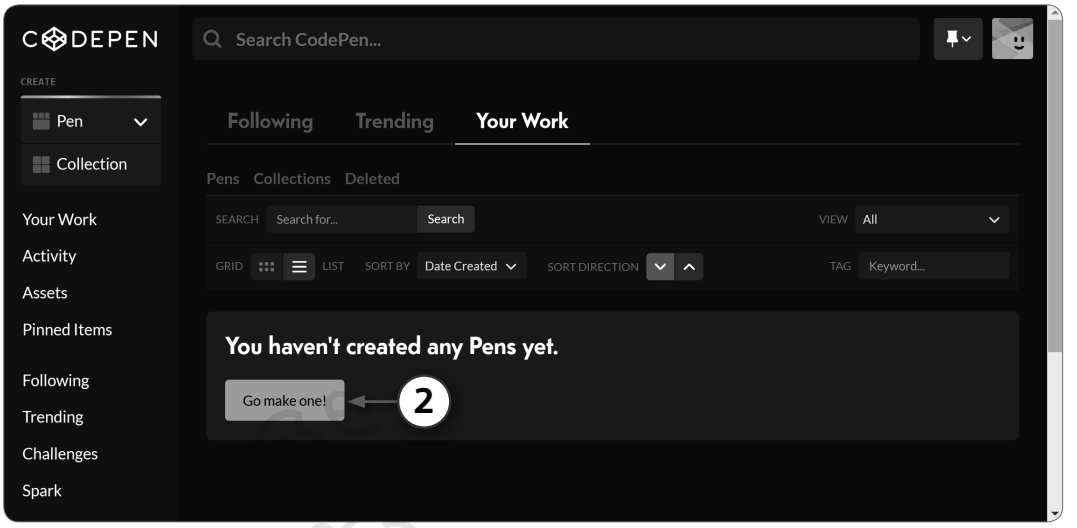
2. ทำตามขั้นตอนต่าง ๆ ตามที่ทางเว็บไซต์แนะนำ เช่น ถ้าสมัครผ่านอีเมลจะต้องเข้าไปคลิกลิงก์ในอีเมลเพื่อยืนยันการสมัครด้วย จากนั้นให้เข้าสู่ระบบให้เรียบร้อย

โค้ดแรกกับ JavaScript

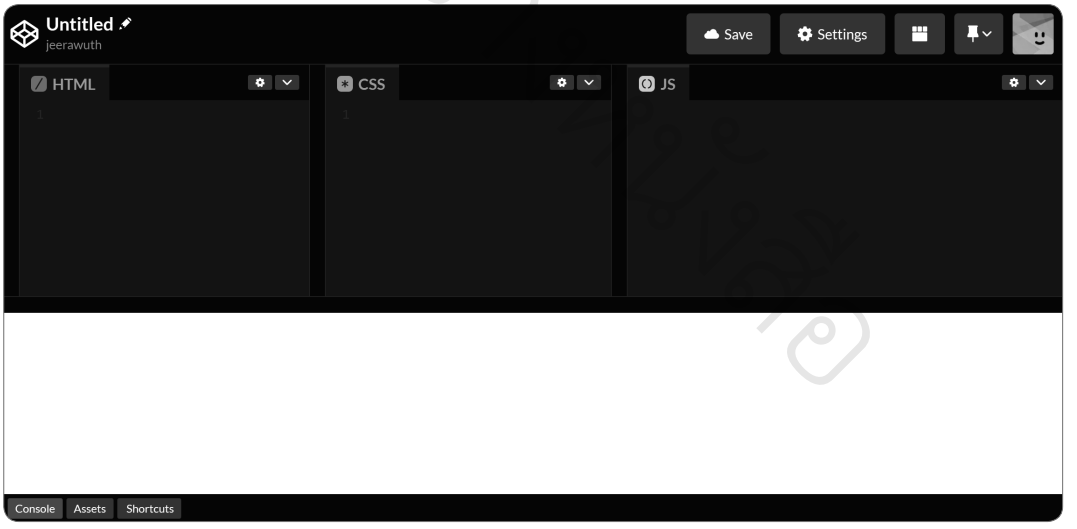
คำสั่งแรกของ JavaScript ที่ต้องทราบคือ คำสั่ง console.log() ใช้เพื่อพิมพ์ข้อความออกมาที่หน้าจอ เช่น เมื่อต้องการพิมพ์ข้อความ "Hello JavaScript" ก็ให้ใช้คำสั่ง console.log("Hello JavaScript") เป็นต้น

1. เปิดเบราว์เซอร์และไปยัง https://codepen.io และล็อกอินเข้าสู่ระบบให้เรียบร้อย จากคลิกเมนู Your Work เพื่อแสดงพื้นที่เก็บงานที่เคยได้ทำเอาไว้ (ในการใช้งานครั้งแรกจะเป็นพื้นที่ว่าง ๆ ยังไม่มีข้อมูลอะไรอยู่)

2. คลิกปุ่ม Go make one! เพื่อเริ่มลงมือเขียนโค้ด



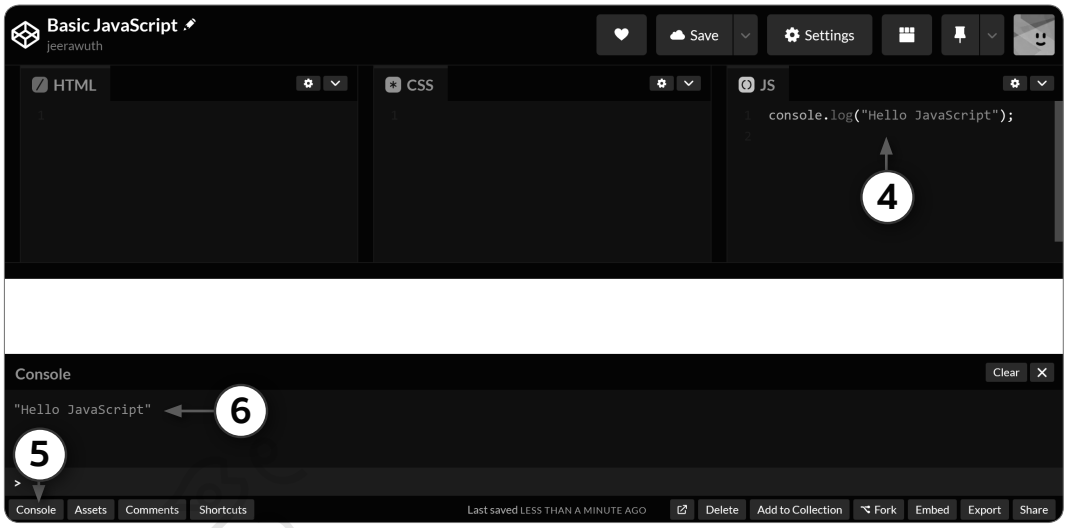
3. ระบบจะกำหนดหน้าต่างขึ้นมาจำนวน 4 หน้าต่าง ด้านบนเรียงจากซ้ายไปขวา คือ หน้าต่าง HTML, CSS และ JS ส่วนด้านล่าง คือ หน้าต่างที่ใช้แสดงผลลัพธ์



4. ที่หน้าต่าง JS ให้กรอกโค้ดภาษา JavaScript ลงไปดังนี้

```
console.log("Hello JavaScript");
```

5. คลิกปุ่ม Console เพื่อแสดงหน้าต่างคอนโซล
6. ผลลัพธ์คำว่า Hello JavaScript จะถูกแสดงที่หน้าต่างคอนโซล ซึ่งเกิดจากคำสั่งที่ได้พิมพ์ไว้ในขั้นตอนที่ 4



7. สามารถตั้งชื่อได้ โดยคลิกปุ่มรูปดินสอ  และแก้ไขชื่อตามต้องการ เช่น Basic JavaScript
8. คลิกปุ่ม Save เพื่อบันทึกโค้ด

Note

โดยปกติเมื่อมีการบันทึกโค้ดครั้งแรกไปแล้ว ถ้ามีการแก้ไขโค้ด CodePen ก็จะบันทึกโค้ดให้อัตโนมัติ ซึ่งผู้ใช้สามารถยกเลิกการบันทึกอัตโนมัตินี้ได้ โดยคลิกลูกศรข้างปุ่ม Save และแก้ไขตัวเลือกตามต้องการ

ตัวอย่าง 1.1
การประกาศตัวแปร

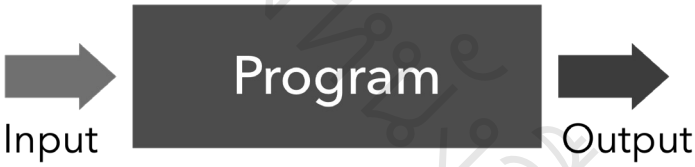
ต่อไปนี้คือตัวอย่างการประกาศตัวแปร (ตั้งชื่อตัวแปร) และกำหนดค่าให้กับตัวแปร

```
1 var first_name = "Sombat";
2 var age = 14;
3 age = 40;
```

- บรรทัดที่ 1 ประกาศตัวแปรโดยใช้ชื่อว่า first_name และกำหนดค่าด้วยข้อความ "Sombat"
- บรรทัดที่ 2 ประกาศตัวแปรโดยใช้ชื่อว่า age และกำหนดค่าตัวแปร age ด้วยเลข 14
- บรรทัดที่ 3 กำหนดค่าตัวแปร age ด้วยตัวเลข 40

อินพุตและเอาต์พุต

แอปพลิเคชันหรือโปรแกรมจะมีพื้นฐานที่สำคัญอยู่ 2 ส่วน คือ อินพุต (Input) และเอาต์พุต (Output) โดยอินพุตคือการที่ผู้ใช้งานโปรแกรมส่งข้อมูลบางอย่างเข้าไปยังโปรแกรม (เช่น กรอกข้อความ คลิกปุ่ม เลื่อนเมาส์ ฯลฯ) ส่วนเอาต์พุต คือ ข้อมูลบางอย่างที่ส่งออกมาจากโปรแกรม (เช่น การแสดงข้อความ การแสดงปุ่ม รูปภาพ ฯลฯ)



▲ รูปแสดงอินพุตและเอาต์พุต

การพัฒนาแอปพลิเคชันจะต้องกำหนดอินพุต เอาต์พุตรูปแบบต่าง ๆ เพื่อใช้ติดต่อสื่อสารระหว่างโปรแกรมกับผู้ใช้ ซึ่งอินพุตขั้นพื้นฐาน ได้แก่ การพิมพ์ข้อความโดยใช้คีย์บอร์ดส่งเข้าไปยังโปรแกรม ส่วนเอาต์พุตขั้นพื้นฐานคือการแสดงข้อความออกมาที่หน้าจอ

รับข้อความที่เป็นอินพุตด้วยคำสั่ง prompt

หากต้องการส่งอินพุตที่เป็นข้อความเข้าไปยังเบราว์เซอร์ สามารถใช้ฟังก์ชัน prompt() ซึ่งการใช้ prompt ใน JavaScript มีรูปแบบพื้นฐาน เป็นดังต่อไปนี้

```
var variable = prompt(message?);
```

- **variable** คือ ตัวแปรที่ใช้เก็บข้อความซึ่งผู้ใช้ได้กรอกลงไป
- **prompt** คือ ฟังก์ชันที่ใช้รับข้อความที่ผู้ใช้กรอกจากคีย์บอร์ด
- **message** คือ ข้อความที่ต้องการนำมาแสดงบนหน้าจอ เพื่อบอกจุดประสงค์ว่าจะนำข้อความที่ผู้ใช้กรอกไปทำอะไร โดยข้อความจะต้องอยู่ภายในเครื่องหมาย Double quote หรือ Single quote เช่น "กรุณากรอกอายุ" หรือ 'กรุณากรอกอายุ' ตามลำดับ

ตัวอย่างการใช้งานฟังก์ชัน prompt

ตัวอย่างการใช้งานฟังก์ชัน prompt() เช่น สั่งให้แสดงข้อความถามชื่อผู้ใช้ว่า "What is your name?" เมื่อผู้ใช้กรอกชื่อแล้วกดปุ่ม <Enter> ชื่อของผู้ใช้ก็จะถูกกำหนดให้กับตัวแปร your_name

1. ไปยังหน้าต่าง JS ของ CodePen
2. กรอกโค้ดคำสั่งเป็นดังต่อไปนี้

```
1 var your_name = prompt("What is your name?");
2 console.log(your_name);
3 var your_age = prompt("Your age is: ");
4 console.log(your_age);
```

- บรรทัดที่ 1 ใช้คำสั่ง prompt() รับข้อความจากผู้ใช้ เมื่อผู้ใช้กรอกชื่อแล้วกดปุ่ม <Enter> ชื่อที่เป็นข้อความจะถูกนำมาเก็บยังตัวแปร your_name
- บรรทัดที่ 2 ใช้คำสั่ง console.log() แสดงข้อความที่อยู่ในตัวแปร your_name ออกมา
- บรรทัดที่ 3 ใช้คำสั่ง prompt() รับข้อความจากผู้ใช้ เมื่อผู้ใช้กรอกอายุแล้วกดปุ่ม <Enter> อายุที่ผู้ใช้กรอกจะถูกนำมาเก็บยังตัวแปร your_age สังเกตว่า แม้ว่าอายุจะเป็นตัวเลข แต่การกรอกตัวเลขด้วยคีย์บอร์ด ตัวเลขนี้จะถือว่าเป็นข้อความชนิดหนึ่ง (ไม่ใช่ตัวเลขที่จะนำมาคำนวณค่าได้)
- บรรทัดที่ 4 ใช้คำสั่ง console.log() แสดงข้อความที่อยู่ในตัวแปร your_age ออกมา

3. หลังจากบันทึกโค้ด จะปรากฏหน้าต่าง "What is your name?" ให้กรอกชื่อ (เช่น Sombat) แล้วคลิกปุ่ม ตกลง

บทที่ 9
พื้นฐานการใช้งานออบเจ็กต์

ออบเจ็กต์ (Object) เป็นชนิดข้อมูลที่มีโครงสร้าง (Data Structure) ชนิดหนึ่งของ JavaScript ที่มีความสำคัญไม่ต่างชนิดข้อมูลพื้นฐานอื่น ๆ ออบเจ็กต์ ถือเป็นชนิดข้อมูลที่กำหนดค่าไปแล้วสามารถแก้ไขค่าได้ (Mutable) จึงสามารถเพิ่มค่า ลบค่า และเปลี่ยนแปลงค่าที่อยู่ในออบเจ็กต์ได้ตามต้องการ

ในบทนี้จะแนะนำพื้นฐานเกี่ยวกับออบเจ็กต์ที่จำเป็นต้องทราบทั้งหมด เริ่มตั้งแต่การกำหนดค่าเริ่มต้นให้กับออบเจ็กต์, การตรวจสอบก่อนแก้ไข หรือนำค่าจากออบเจ็กต์มาใช้งาน และปัญหาที่มักพบเกี่ยวกับการใช้งานออบเจ็กต์

ออบเจ็กต์ใน JavaScript

การเก็บข้อมูลในทางปฏิบัติ ไม่ได้เก็บเป็นค่าเดี่ยว ๆ แต่มักอยู่ในรูปแบบที่ซับซ้อน เช่น การเก็บข้อมูลนักเรียนจะมีชื่อ นามสกุล รหัสนักเรียน ชั้นเรียน ครูประจำชั้น ที่อยู่ คะแนนสอบ ฯลฯ

JavaScript สามารถนำชุดของข้อมูล มารวมกันเป็นกลุ่มข้อมูล เรียกว่า ออบเจ็กต์ (Object) ภายในออบเจ็กต์สามารถกำหนดโครงสร้างข้อมูลในแบบต่าง ๆ ได้

ออบเจ็กต์ เป็นชนิดข้อมูลแบบหนึ่ง ซึ่งเป็นการจับคู่กันระหว่าง key (คีย์) และ value (ค่า) ซึ่ง key ที่อยู่ในออบเจ็กต์จะต้องไม่ซ้ำกัน คล้ายกับการจับคู่กันระหว่างคำศัพท์ และคำแปลคำศัพท์ในพจนานุกรม ที่คำศัพท์จะไม่ซ้ำ แต่ความหมายของคำศัพท์สามารถซ้ำกันได้ เป็นต้น

รูปแบบของ ออบเจ็กต์ จะเป็นดังต่อไปนี้

```
{ key : value }
```

- **key** เป็นชนิดข้อมูลแบบใดก็ได้ที่ไม่ซ้ำกับ key ตัวอื่น ๆ ในออบเจ็กต์เดียวกัน
- **value** คือ ค่าที่จะกำหนดให้กับ key ซึ่งจะเป็นข้อมูลชนิดใดก็ได้ และสามารถซ้ำกับ value อื่น ๆ ได้

พร็อพเพอร์ตี (Property)

ภายในออบเจ็กต์ สามารถกำหนดชื่อชุดข้อมูลขึ้นใช้งานเองได้ โดยกำหนดเป็นคู่ระหว่าง ชื่อข้อมูล (key) และค่าที่จะกำหนดให้กับข้อมูลนั้น ๆ (value) เช่น กำหนด name เพื่อเก็บชื่อนักเรียน และกำหนด age เพื่อเก็บอายุนักเรียน

เรียกชื่อที่ใช้อธิบายชุดข้อมูลที่อยู่ภายในออบเจ็กต์ว่า พร็อพเพอร์ตี (Property) จากตัวอย่าง name ก็คือพร็อพเพอร์ตี และ age ก็เป็นพร็อพเพอร์ตีเช่นเดียวกัน

```
1 {
2     name: "Sombat",
3     age: 14,
4 };
```

- บรรทัดที่ 1 กำหนดเครื่องหมาย { เพื่อกำหนดจุดเริ่มของออบเจ็กต์
- บรรทัดที่ 2 กำหนดชื่อพร็อพเพอร์ตี name และกำหนดค่าสตริง "Sombat" ให้กับพร็อพเพอร์ตี name
- บรรทัดที่ 3 กำหนดชื่อพร็อพเพอร์ตี age และกำหนดตัวเลข 14 ให้กับพร็อพเพอร์ตี age
- บรรทัดที่ 4 ใส่เครื่องหมาย } เพื่อกำหนดจุดสิ้นสุดของออบเจ็กต์

พร็อพเพอร์ตีจะประกอบด้วย 2 ส่วนที่เป็นคู่กัน ได้แก่ key และ value โดย key คือ ชื่อพร็อพเพอร์ตีที่ต้องไม่ซ้ำกัน ส่วน value คือ ค่าที่กำหนดให้กับพร็อพเพอร์ตี

Note

พร็อพเพอร์ตี (key) ที่อยู่ภายในออบเจ็กต์ จะมีชนิดข้อมูลเป็นแบบ String โดยอัตโนมัติ ส่วน value จะเป็นชนิดข้อมูลใดก็ได้ เช่น เป็น String, Number และอื่น ๆ ก็ได้

การกำหนดออบเจ็กต์เปล่า

ออบเจ็กต์เปล่า คือ ออบเจ็กต์ที่ยังไม่ได้กำหนดพร็อพเพอร์ตีใด ๆ ลงไป ใน JavaScript การกำหนดออบเจ็กต์เปล่า จะใช้เครื่องหมาย { } หรือใช้คำสั่ง new Object() ดังตัวอย่าง

```
1 const firstObject = { };
2 const secondObject = new Object();
3 console.log(typeof firstObject);    // object
4 console.log(typeof secondObject);   // object
```

- บรรทัดที่ 1 กำหนดออบเจ็กต์เปล่า ไปยังตัวแปร firstObject โดยใช้ { }
- บรรทัดที่ 2 กำหนดออบเจ็กต์เปล่า ไปยังตัวแปร secondObject โดยใช้คำสั่ง new Object()
- บรรทัดที่ 3 เมื่อใช้ typeof โอเปอเรเตอร์ตรวจสอบชนิดของตัวแปร firstObject จะได้ผลลัพธ์เป็นสตริง 'object'

ตัวอย่าง 9.1
กำหนดออบเจ็กต์ให้กับตัวแปร

การกำหนดออบเจ็กต์ สามารถกำหนดค่าเริ่มต้นให้กับพร็อพเพอร์ตีต่าง ๆ ที่อยู่ภายในออบเจ็กต์ ผู้ใช้สามารถตั้งชื่อพร็อพเพอร์ตีขึ้นใช้งานเอง และกำหนดค่าไปยังพร็อพเพอร์ตีด้วยชนิดข้อมูลแบบต่าง ๆ ได้ เช่น String, Number หรือชนิดข้อมูลแบบอื่น ๆ

- บรรทัดที่ 6 ตั้งชื่อตัวแปร p แทนพร็อพเพอร์ตี price และตั้งชื่อ pName แทนพร็อพเพอร์ตี productName

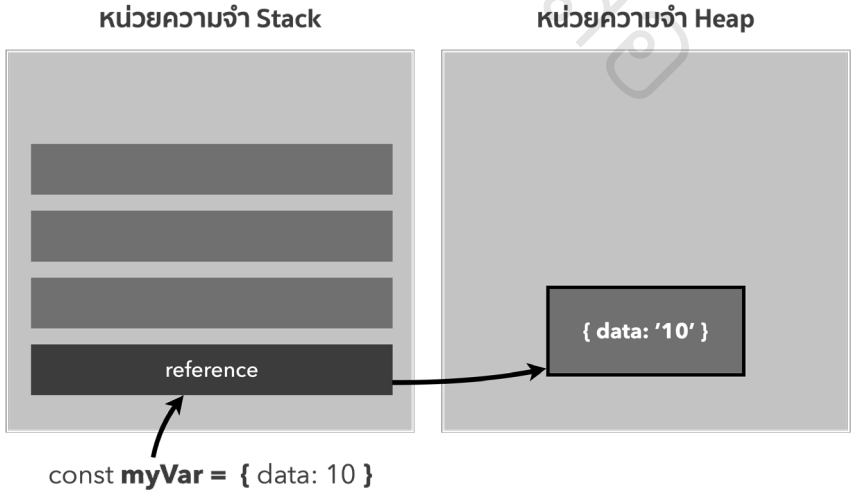
การประกาศตัวแปรเพื่อรับค่าจากออบเจ็กต์ สามารถประกาศตัวแปรเพื่อเก็บพร็อพเพอร์ตีที่เหลือได้ โดยใช้ Spread Operator (...) ตามด้วยชื่อตัวแปร เช่น ...other (ตั้งชื่อตัวแปรซึ่งไม่ตรงกับชื่อพร็อพเพอร์ตี) เพื่อกำหนดออบเจ็กต์สำหรับเก็บค่าพร็อพเพอร์ตีที่เหลือทั้งหมด

```
1 const myObject = {
2     id: '1234',
3     productName: 'Mee Pro X 2023',
4     price: 299
5 };
6 let {productName, ...other} = myObject;
7 console.log(productName);   // Mee Pro X 2023
8 console.log(other);         // { id: '1234', price: 299 }
```

- บรรทัดที่ 6 ประกาศตัวแปร productName เก็บค่าพร็อพเพอร์ตี productName และประกาศตัวแปร other เพื่อเก็บออบเจ็กต์ โดยพร็อพเพอร์ตีที่ยังไม่ได้ประกาศตัวแปรมารับค่า จะถูกนำมาใส่ยังออบเจ็กต์ other นี้ (พร็อพเพอร์ตี id และ พร็อพเพอร์ตี price)

ออบเจ็กต์เป็น Reference Value

ออบเจ็กต์มีลักษณะเป็น Reference Value หมายถึง ตัวแปรไม่ได้เก็บข้อมูลที่อยู่ในออบเจ็กต์ทั้งหมดโดยตรง แต่จะใช้วิธีเก็บตัวอ้างอิง (Reference) เพื่อชี้ไปยังข้อมูลที่อยู่ในหน่วยความจำอีกต่อหนึ่ง ดังนั้น เมื่อกำหนดออบเจ็กต์ให้กับตัวแปร จึงเป็นการกำหนดค่าอ้างอิง (Reference) ให้กับตัวแปร ไม่ใช่กำหนดข้อมูลจริงให้กับตัวแปร



บทที่ 15
การสืบทอดคลาส (Class Inheritance)

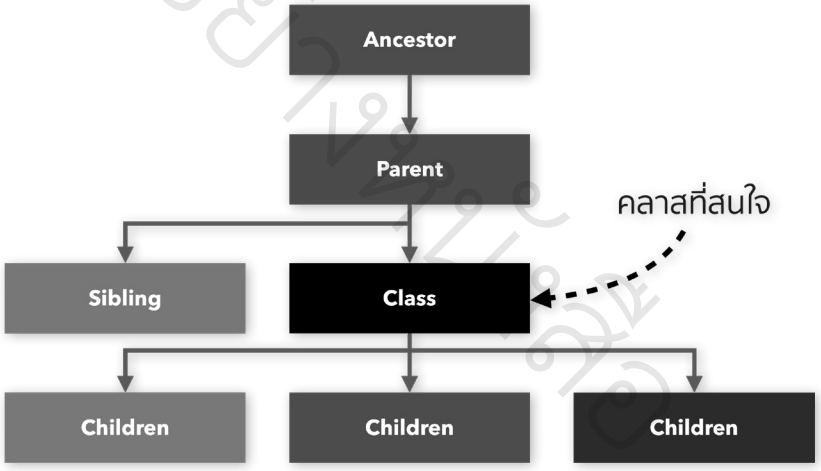
ในบทนี้จะอธิบายเกี่ยวกับพื้นฐานการสืบทอดคลาส (class inheritance) เริ่มตั้งแต่วิธีกำหนดลำดับชั้นของคลาส, การตรวจสอบคลาสในลำดับชั้น, วิธีสืบทอดเมธอดและพร็อพเพอร์ตีมาจากคลาสอื่น, การกำหนด constructor ด้วยรูปแบบที่แตกต่างกัน, การเพิ่มคุณสมบัติใหม่ ๆ ให้กับคลาสลูก (extend), การแก้ไขเมธอดที่สืบทอดมา (override), การใช้งานเมธอดที่เป็นของ superclass เป็นต้น

การสืบทอดคลาส (class inheritance)

ออบเจ็กต์ที่สร้างจากคลาสที่ต่างกัน ก็จะมีคุณสมบัติหรือความสามารถที่ต่างกัน แต่ในบางครั้งบางออบเจ็กต์ก็อาจมีคุณสมบัติที่คล้ายกัน และก็อาจมีความสามารถบางอย่างที่เหมือนกัน

เพื่อให้ได้ออบเจ็กต์ที่มีคุณสมบัติบางส่วนร่วมกัน โดยไม่ต้องเขียนโค้ดซ้ำ ๆ จึงนำ class มาจัดเรียงเป็นลำดับชั้น เพื่อแชร์คุณสมบัติหรือส่งต่อความสามารถบางอย่างไปให้คลาสอื่น ๆ ที่อยู่ภายใต้คลาสนั้น ๆ

เรียกการถ่ายทอดคุณสมบัติหรือความสามารถจากคลาสหนึ่งไปให้กับอีกคลาสตามลำดับชั้นเช่นนี้ว่า การสืบทอดคลาส (class inheritance)

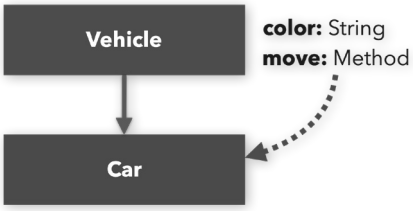


จากรูปด้านบน คือ ตัวอย่างการนำคลาส (class) มาเรียงต่อกันเป็นลำดับชั้น โดยมีคำศัพท์สำหรับใช้อ้างอิงคลาสต่าง ๆ ที่อยู่ในลำดับชั้น ดังนี้

- คลาสที่อยู่ในระดับซึ่งเหนือกว่า และอยู่ติดกับคลาสที่สนใจ เรียกว่า superclass หรือ parent (คลาสแม่)
- คลาสที่อยู่ภายใต้ และติดกับคลาสที่กำลังสนใจ เรียกว่า subclass หรือ children (คลาสลูก)
- คลาสที่อยู่ในระดับเหนือกว่า แต่ไม่ติดกันคลาสที่สนใจ เรียกว่า ancestor (คลาสที่เป็นบรรพบุรุษ)
- class ที่มี parent เดียวกันกับ class ที่สนใจในปัจจุบัน แต่ไม่ได้สืบทอดต่อจากคลาสปัจจุบัน เรียกว่า sibling

ตัวอย่าง 15.1
กำหนดคลาส Car สืบทอดต่อจาก Vehicle

ต่อไปนี้เป็นตัวอย่างการกำหนดคลาส Car ให้สืบทอดต่อจากคลาส Vehicle ทำให้พร็อพเพอร์ตี และเมธอดต่าง ๆ ที่อยู่ใน Vehicle จะถูกส่งต่อหรือถูกถ่ายทอดไปให้กับคลาส Car โดยอัตโนมัติ

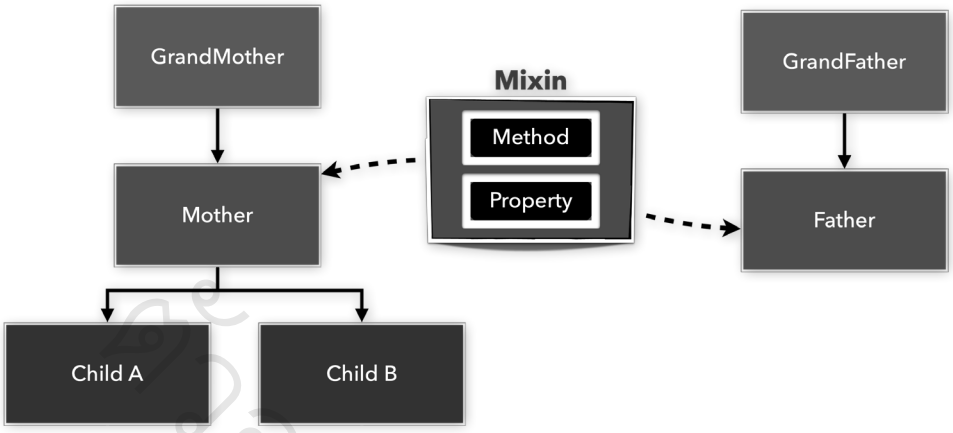



```
1  class Vehicle {
2      constructor() {
3          this.color = 'red'
4      }
5      move() {
6          console.log("I can move!!!")
7      }
8  }
9  class Car extends Vehicle{
10     // empty statement
11 }
12
13 const myCar = new Car()
14 myCar.move()                    // I can move!!!
15 console.log(myCar.color)        // red
```

- บรรทัดที่ 1 ประกาศคลาส Vehicle
- บรรทัดที่ 3 กำหนดค่า 'red' ให้กับพร็อพเพอร์ตี color
- บรรทัดที่ 5 กำหนดเมธอด move()
- บรรทัดที่ 9 ประกาศคลาส Car โดยสืบทอดต่อจากคลาส Vehicle ทำให้พร็อพเพอร์ตี color และเมธอด move() ที่อยู่ในคลาส Vehicle ถูกถ่ายทอดมายังคลาส Car โดยอัตโนมัติ
- บรรทัดที่ 13 ประกาศตัวแปร myCar เก็บค่าออบเจ็กต์ ที่สร้างจากคลาส Car
- บรรทัดที่ 14 เรียกใช้เมธอด move() ซึ่งเป็นเมธอดที่สืบทอดมาจากคลาส Vehicle
- บรรทัดที่ 15 พิมพ์ค่าที่เก็บอยู่ในพร็อพเพอร์ตี color ซึ่งเป็นพร็อพเพอร์ตีที่สืบทอดมาจากคลาส Vehicle

รู้จักกับ Mixin

Mixin คือ คลาสที่ได้กำหนดเมธอดหรือพร็อพเพอร์ตีเอาไว้ ซึ่งคลาสอื่น ๆ สามารถนำเมธอดและพร็อพเพอร์ตีที่อยู่ใน Mixin ไปใช้งานได้ โดยไม่ต้องสืบทอดต่อจาก Mixin



จากรูปคลาส Mother และคลาส Father อยู่คนละแขนงกัน ไม่ได้มีการสืบทอดต่อจากกัน แต่คลาส Mother สามารถใช้งานเมธอดและพร็อพเพอร์ตีที่อยู่ในคลาส Mixin และคลาส Father ก็สามารถใช้งานเมธอดและพร็อพเพอร์ตีจาก Mixin ได้ การใช้ Mixin จึงเป็นอีกวิธีหนึ่ง ที่ใช้แชร์คุณสมบัติที่ถูกใช้งานซ้ำ ๆ ไปยังคลาสอื่นได้

การกำหนด Mixin ใน JavaScript จะใช้วิธีกำหนดออบเจ็กต์ไปยัง Prototype ของคลาส ดังนี้

```
1  const MyMixin = {
2      mixinProperty: 0,
3      mixinMethod() {
4          // statement
5      }
6  }
7  class MyClass {
8      // statement
9  }
10 Object.assign(MyClass.prototype, MyMixin);
```

- บรรทัดที่ 1 ประกาศตัวแปร MyMixin เก็บข้อมูลในแบบออบเจ็กต์
- บรรทัดที่ 2 กำหนดพร็อพเพอร์ตี mixinProperty ภายใน MyMixin
- บรรทัดที่ 3-5 กำหนดเมธอด mixinMethod ภายใน MyMixin
- บรรทัดที่ 7 กำหนด MyClass
- บรรทัดที่ 10 กำหนด MyMixin ไปยัง prototype ของคลาส MyClass เพื่อทำให้ออบเจ็กต์ที่สร้างจาก MyClass สามารถใช้งานพร็อพเพอร์ตี และเมธอดที่อยู่ใน MyMixin ได้

บทที่ 18
การใช้ JavaScript จัดการกับ DOM

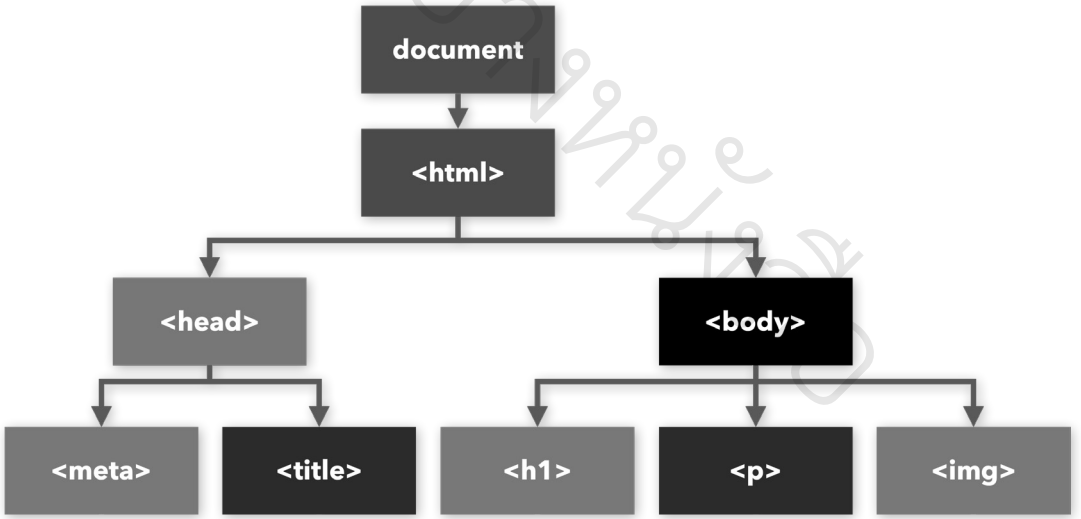
ในบทนี้จะเป็นการนำ JavaScript มาใช้เพื่อจัดการกับเว็บเพจ โดยเริ่มจากแนะนำพื้นฐานเกี่ยวกับ DOM จากนั้นจึงใช้คำสั่ง JavaScript เพื่อเข้าไปยัง DOM และแก้ไขส่วนต่าง ๆ ของเว็บเพจตามต้องการ เช่น การเลือกเอลิเมนต์ในเว็บเพจ การแก้ไขข้อความบนเว็บเพจ การตกแต่งเว็บเพจ ฯลฯ

รู้จักกับ DOM

เว็บเพจในอดีตจะมีลักษณะคงที่ (Static) เมื่อโหลดหน้าเว็บเพจมาแล้วเนื้อหาจะเปลี่ยนแปลงไม่ได้ ไม่สามารถโต้ตอบกับผู้ใช้ได้ หากต้องการแก้ไขเนื้อหาเว็บเพจจะต้องโหลดเนื้อหาขึ้นมาใหม่ทั้งหมด

เพื่อให้เว็บเพจสามารถเปลี่ยนแปลงเนื้อหา และสามารถโต้ตอบกับผู้ใช้ได้ (Dynamic) จึงได้แยกส่วนประกอบต่าง ๆ บนเว็บเพจออกเป็นออบเจ็กต์ย่อย ๆ และนำมาเรียงต่อกันเป็นโครงสร้างที่เรียกว่า DOM (Document Object Model) การกำหนด DOM นี้จะทำให้สามารถใช้ JavaScript เพื่อแก้ไขหรือปรับแต่งเอลิเมนต์ต่าง ๆ บนเว็บเพจได้ง่าย

เมื่อเบราว์เซอร์โหลด HTML และ CSS ก็จะสร้าง JavaScript ออบเจ็กต์ขึ้น จากนั้นจึงนำไปจัดโครงสร้างกำหนดไว้บน DOM ดังรูป



▲ รูปแสดงตัวอย่างเว็บเพจ และโครงสร้างที่อยู่ใน DOM

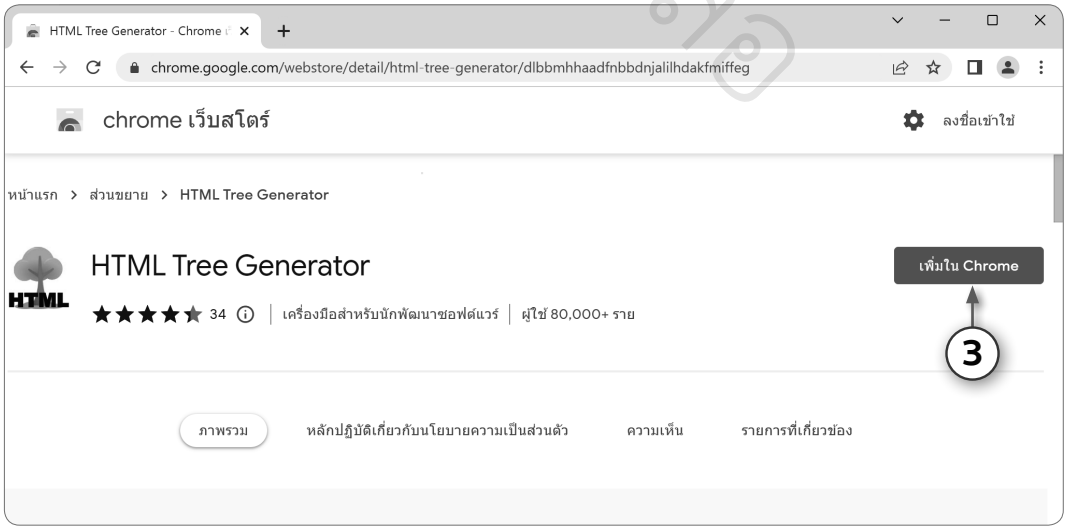
การกำหนด DOM ช่วยให้โปรแกรมเมอร์สามารถเขียน JavaScript เพื่อดูค่า หรือปรับแต่งแก้ไขส่วนต่าง ๆ บนเว็บเพจได้ เช่น ใช้ JavaScript เปลี่ยนสีข้อความ, ใช้ JavaScript แสดงหน้าต่างล็อกอินให้ลอยเหนือส่วนอื่น ๆ, ใช้ JavaScript ตรวจสอบความยาวของรหัสผ่าน, ใช้ JavaScript ตรวจสอบว่าผู้ใช้กรอกข้อมูลในแบบฟอร์มครบและถูกต้อง, ใช้ JavaScript สร้างแอนิเมชันบนเว็บเพจ ฯลฯ

ตรวจสอบโครงสร้างของเว็บเพจ

หากต้องการทราบว่า เว็บเพจที่สนใจนั้นมีการกำหนดโครงสร้างเป็นอย่างไร ให้ติดตั้งส่วนขยาย (Extension) ที่ชื่อว่า HTML Tree Generator ลงในบน Google Chrome ดังนี้

1. เปิดเบราว์เซอร์ Google Chrome จากไปยังเว็บ Google Web Store (ที่ลิงก์ https://chrome.google.com/webstore/category/extensions)
2. ค้นหาส่วนประกอบเสริมที่ต้องการติดตั้ง โดยในที่นี้คือ HTML Tree Generator

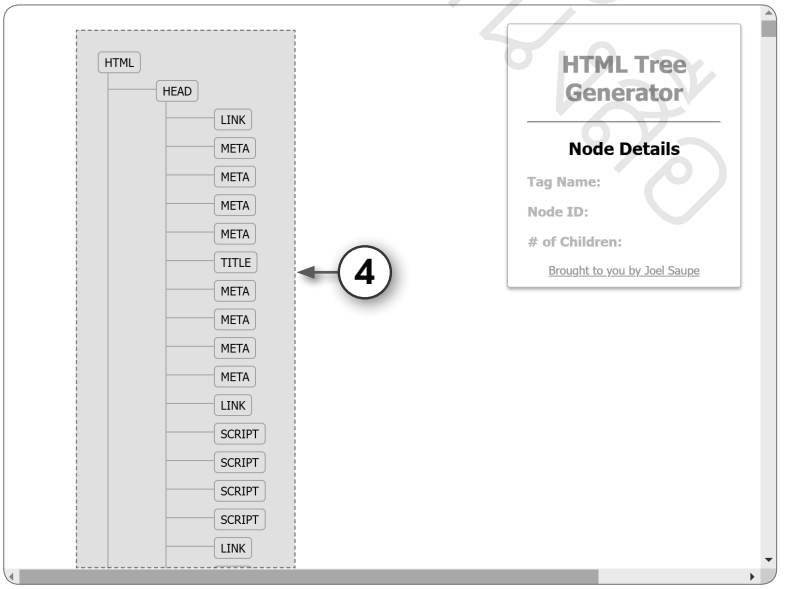
3. ติดตั้งส่วนขยายโดยคลิกปุ่ม เพิ่มใน Chrome



เมื่อต้องการตรวจสอบโครงสร้างของเว็บเพจ ก็ให้เปิดเบราว์เซอร์ Google Chrome ไปยังเว็บเพจที่ต้องการ และสั่งให้แสดงโครงสร้างที่อยู่ในเว็บเพจนั้น ๆ ดังตัวอย่าง

1. ใช้ Google Chrome ไปยังเว็บเพจที่ต้องการ เช่น https://thinkbeyondbook.com
2. คลิกปุ่ม  เพื่อแสดงรายชื่อส่วนขยายที่เปิดใช้งานอยู่

3. คลิกที่ HTML Tree Generator
4. จะพบโครงสร้างของเว็บเพจในแบบแผนภูมิต้นไม้



แพทเทิร์นของ Selector ที่ควรทราบ

การกำหนด Selector เพื่อกำหนดเงื่อนไขในการเลือกเอลิเมนต์มาจาก DOM สามารถใช้ไอดี, ชื่อคลาส, ชื่อแท็ก หรือใช้ค่าที่อยู่ในแอตทริบิวต์ต่าง ๆ ภายในแท็กก็ได้

เลือกเอลิเมนต์ตามค่า id

เลือกเอลิเมนต์โดยอ้างอิงจากค่าที่กำหนดให้กับแอตทริบิวต์ id จะใช้เครื่องหมาย # นำหน้าชื่อไอดี เช่น #my-id หรือ #your-id

Selector	ตัวอย่าง	คำอธิบาย
#id	#main	เลือกเอลิเมนต์ที่ได้กำหนดค่า main ไปยังแอตทริบิวต์ id เช่น

เลือกเอลิเมนต์ตามชื่อคลาส

เลือกเอลิเมนต์โดยใช้ชื่อคลาสจะใช้จุด (.) นำหน้าชื่อคลาส เช่น .my-class หรือ .your-class

Selector	ตัวอย่าง	คำอธิบาย
.class	.menu-item	เลือกเอลิเมนต์ทั้งหมดที่ได้กำหนดค่า menu-item ไปยังแอตทริบิวต์ class เช่น
.class1.class2	.menu-item.active	เลือกเอลิเมนต์ทั้งหมดที่กำหนดชื่อคลาส menu-item และ active ไปยังแอตทริบิวต์ class เช่น

เลือกเอลิเมนต์โดยใช้ชื่อแท็ก

การเลือกเอลิเมนต์โดยใช้ชื่อแท็ก สามารถกำหนดด้วยชื่อแท็กเพียงชื่อเดียว หรือเลือกเอลิเมนต์จากหลาย ๆ แท็กพร้อมกันก็ได้ นอกจากนั้นยังสามารถกำหนดเงื่อนไขร่วมกับการใช้ชื่อคลาสร่วมด้วยก็ได้

Selector	ตัวอย่าง	คำอธิบาย
tag	p	เลือกเอลิเมนต์ที่เป็นแท็ก p ทั้งหมด
tag_1, tag_2	li, a	เลือกเอลิเมนต์ที่เป็นแท็ก li และแท็ก p ทั้งหมด
tag.class	div.container	เลือกเอลิเมนต์ที่เป็นแท็ก div และมีคลาสชื่อว่า container เช่น

เลือกเอลิเมนต์ตามความสัมพันธ์ใน DOM

เลือกเอลิเมนต์โดยใช้โครงสร้างการจัดเรียงเอลิเมนต์ใน DOM เช่น เลือกแท็ก p ทั้งหมดที่อยู่ภายใต้แท็ก div

Selector	ตัวอย่าง	คำอธิบาย
tag	p	เลือกเอลิเมนต์ที่เป็นแท็ก p ทั้งหมด
tag_1 tag_2	div p	เลือกเอลิเมนต์ที่เป็นแท็ก p ทั้งหมดซึ่งอยู่ภายใต้แท็ก div
tag_1 > tag_2	div > p	เลือกเอลิเมนต์ที่เป็นแท็ก p ทั้งหมดซึ่งอยู่ภายใต้แท็ก div เฉพาะในลำดับชั้นที่ติดกัน (parent กับ child)
tag_1 + tag_2	div + p	เลือกเอลิเมนต์ทั้งหมดที่เป็นแท็ก p เอลิเมนต์แรก ที่อยู่ติดกับแท็ก div
tag_1 ~ tag_2	div ~ p	เลือกเอลิเมนต์ที่เป็นแท็ก p ทั้งหมด วางอยู่ด้านหลังแท็ก div ภายในลำดับชั้นเดียวกัน (แท็ก p ไม่จำเป็นต้องอยู่ติดกับแท็ก div)